# Advanced malware in 2023. How prepared are you?


by André Lima (0x4ndr3), at Sikkerhetsfestivalen - 2023

## > whoami

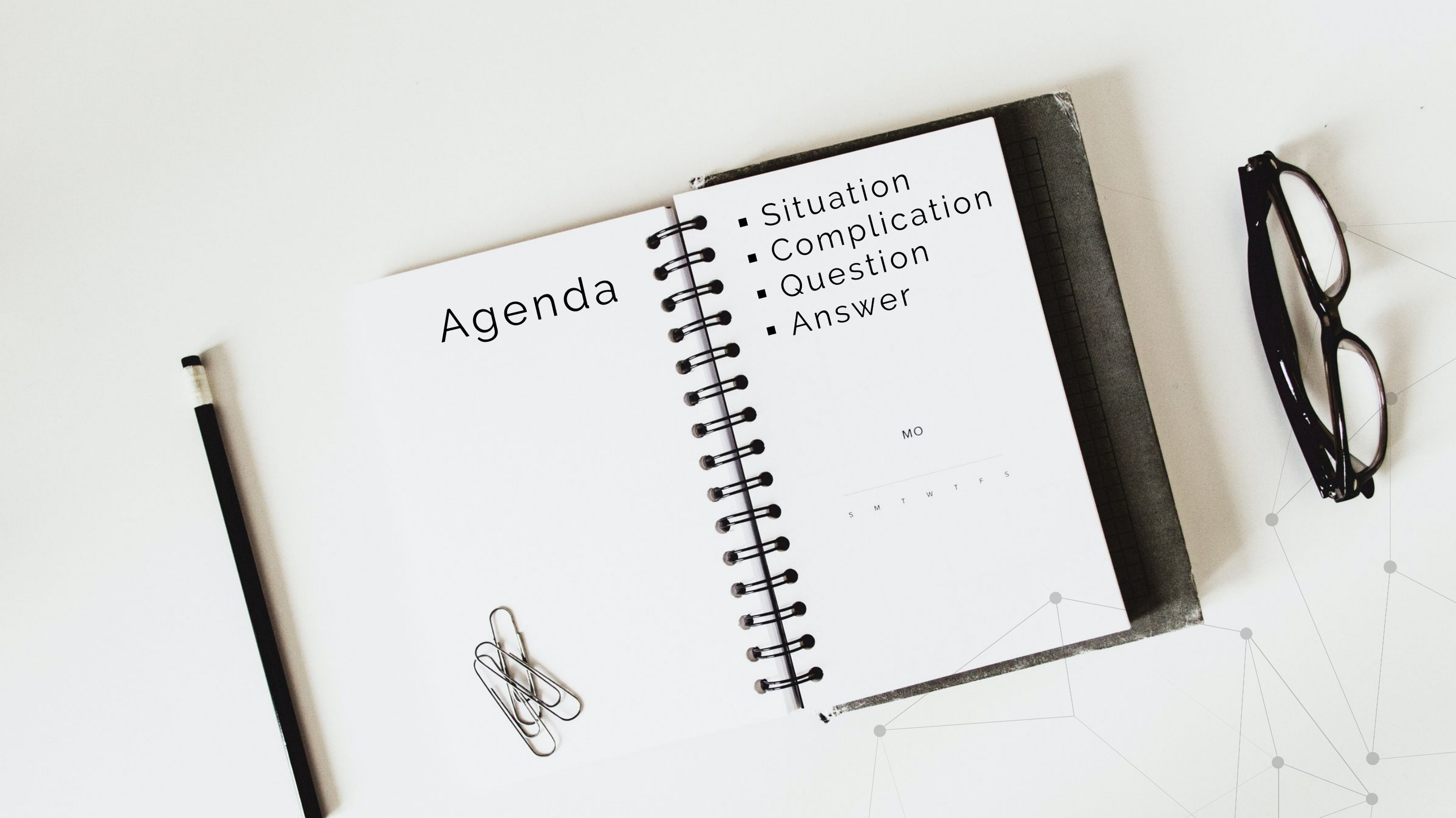
- Work @ PwC Norway
- 10+y: Pentester | Red Team Operator | Researcher
- Worked in Portugal (Lisbon), Australia (Melbourne), and now Norway (Oslo)
- Blogger + Youtube channel
- Certs: OSED, eCRE, SLAE64, etc
- Best friends: windbg, IDA, assembly

https://www.linkedin.com/in/aflima/

Conferences: https://github.com/0x4ndr3/Presentations

https://www.youtube.com/@0x4ndr3

0x4ndr3


Agenda
Situation
Complication
Question
Answer
MO
S M T W T F S

# Situation

Stages of Information security management: 3rd party security service procurement

- Security Architecture review

- Configuration reviews

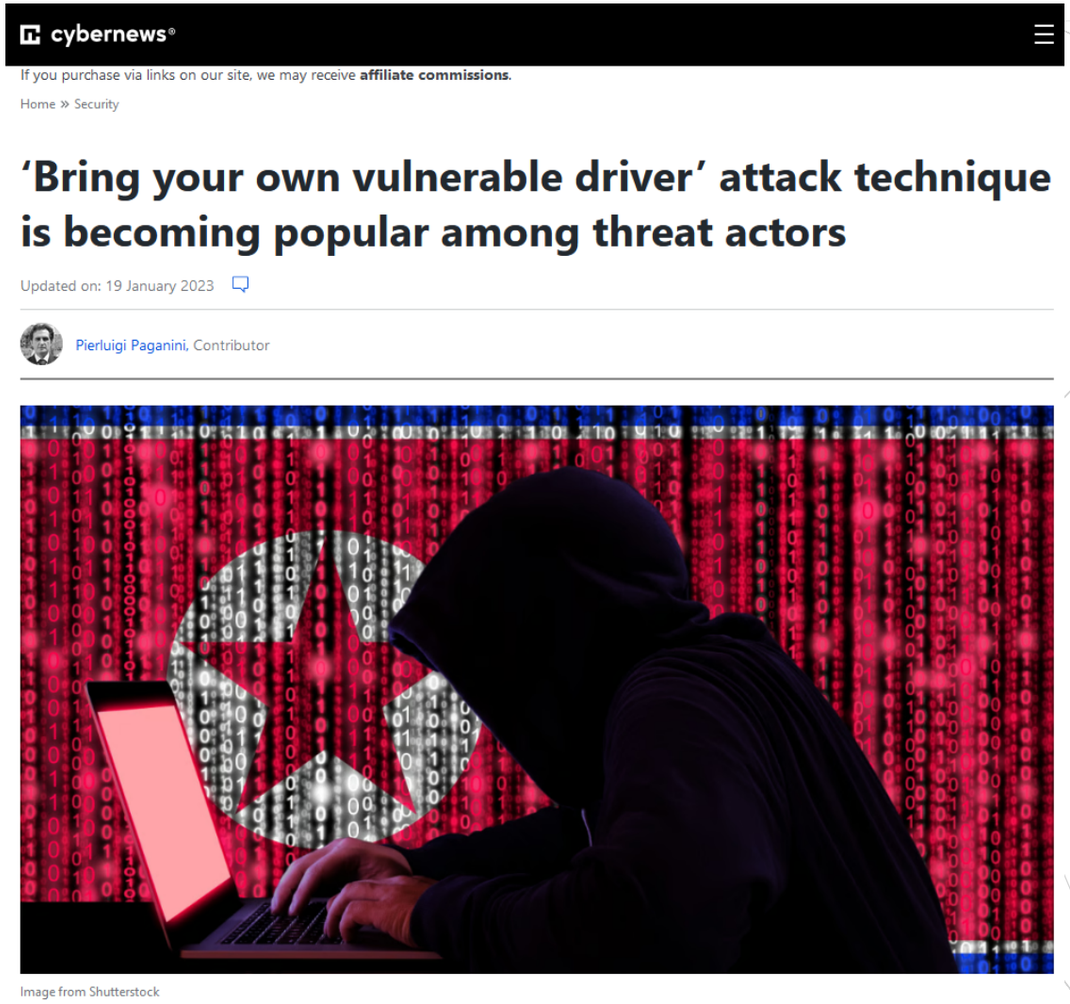
- Penetration Tests (Pentests)

- Red Teams & Purple Teams

**Situation**
Complication
Question
Answer

# Complication

State-sponsored attacks:
- Ransomware



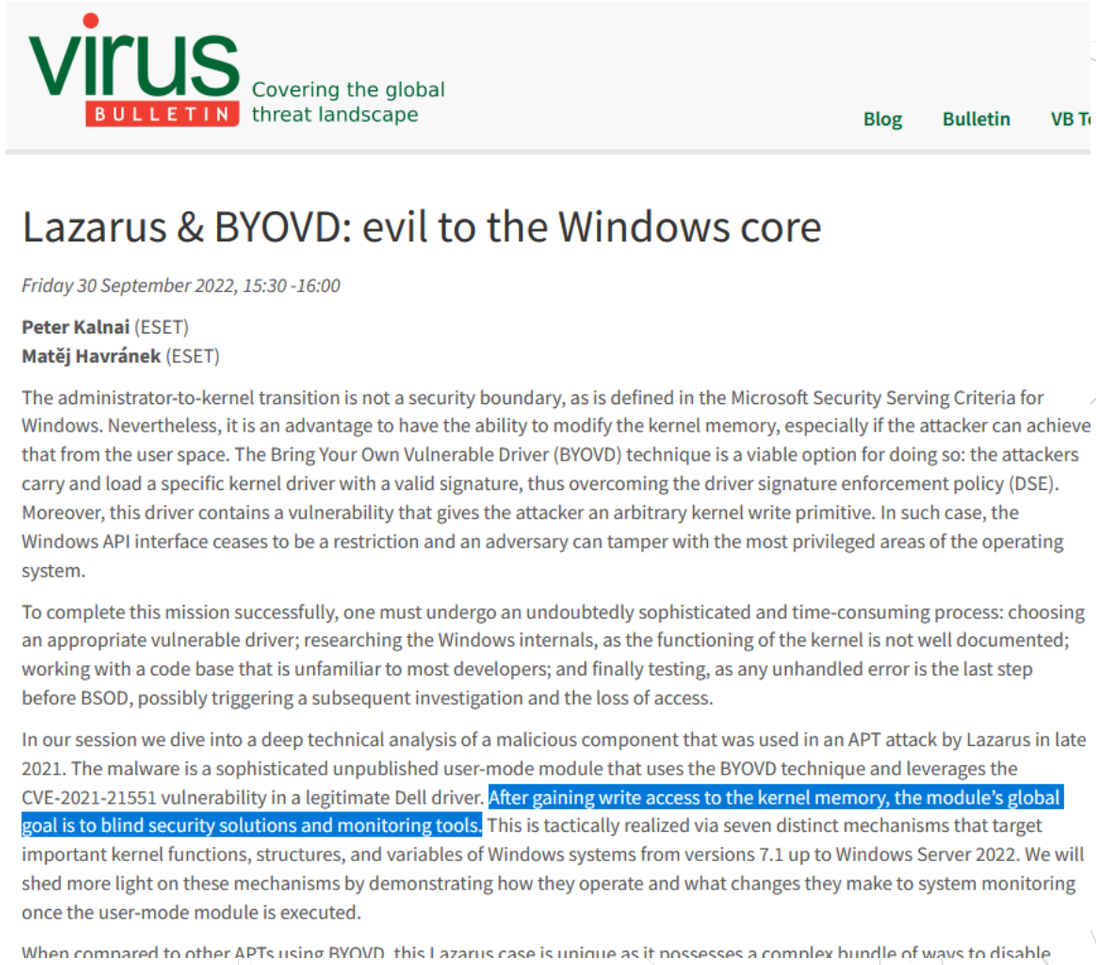

Source: https://cybernews.com/security/bring-your-own-vulnerable-driver-attack/

Situation
**Complication**
Question
Answer

Situation
**Complication**
Question
Answer

# Complication

State-sponsored attacks:
- Ransomware
- Blind security solutions



virus BULLETIN Covering the global threat landscape

Blog Bulletin VB T

## Lazarus & BYOVD: evil to the Windows core

*Friday 30 September 2022, 15:30 -16:00*

**Peter Kalnai** (ESET)
**Matěj Havránek** (ESET)

The administrator-to-kernel transition is not a security boundary, as is defined in the Microsoft Security Serving Criteria for Windows. Nevertheless, it is an advantage to have the ability to modify the kernel memory, especially if the attacker can achieve that from the user space. The Bring Your Own Vulnerable Driver (BYOVD) technique is a viable option for doing so: the attackers carry and load a specific kernel driver with a valid signature, thus overcoming the driver signature enforcement policy (DSE). Moreover, this driver contains a vulnerability that gives the attacker an arbitrary kernel write primitive. In such case, the Windows API interface ceases to be a restriction and an adversary can tamper with the most privileged areas of the operating system.

To complete this mission successfully, one must undergo an undoubtedly sophisticated and time-consuming process: choosing an appropriate vulnerable driver; researching the Windows internals, as the functioning of the kernel is not well documented; working with a code base that is unfamiliar to most developers; and finally testing, as any unhandled error is the last step before BSOD, possibly triggering a subsequent investigation and the loss of access.

In our session we dive into a deep technical analysis of a malicious component that was used in an APT attack by Lazarus in late 2021. The malware is a sophisticated unpublished user-mode module that uses the BYOVD technique and leverages the CVE-2021-21551 vulnerability in a legitimate Dell driver. After gaining write access to the kernel memory, the module's global goal is to blind security solutions and monitoring tools. This is tactically realized via seven distinct mechanisms that target important kernel functions, structures, and variables of Windows systems from versions 7.1 up to Windows Server 2022. We will shed more light on these mechanisms by demonstrating how they operate and what changes they make to system monitoring once the user-mode module is executed.

When compared to other APTs using BYOVD, this Lazarus case is unique as it possesses a complex bundle of ways to disable

Source: https://www.virusbulletin.com/conference/vb2022/abstracts/lazarus-byovd-evil-windows-core/

# Complication

Demo 1 / 3
https://youtu.be/WJq_6a7fKAM

Situation
**Complication**
Question
Answer

# Complication

Demo 2 / 3
https://youtu.be/gliF3yRD6sM

Situation
**Complication**
Question
Answer

# Complication

Demo 3 / 3
https://youtu.be/pGCMIJEpmaY

Situation
**Complication**
Question
Answer

# Question

Am I prepared for this?
How do I test myself?

Situation
Complication
**Question**
Answer

Situation
Complication
Question
**Answer**

# Answer

- Step #1 (attack): loading the rootkit/driver
  - EDR presence
  - Windows security features
    - Use modern OS versions
    - HVCI
    - WDAC
  - Relevant event ID

But still...

- Not all vulnerable drivers are in Microsoft's "list"
- Attackers can sign their own drivers
- 0-days

Situation
Complication
Question
**Answer**

# Answer

- Step #2 (attack): malicious kernel ops
  - Know your environment: which drivers are supposed to be loaded? Can you tell the difference?

Administrator: Windows PowerShell

```
PS C:\Windows\system32> Get-WindowsDriver -Online -All | Select-Object Driver,OriginalFileName,ProviderName

Driver                                  OriginalFileName

------                                  ----------------
1394.inf                                C:\Windows\System32\DriverStore\FileRepository\1394.inf_amd64_aee05b5c33eee
3ware.inf                               C:\Windows\System32\DriverStore\FileRepository\3ware.inf_amd64_408ceed6ec8a
61883.inf                               C:\Win
acpi.inf                                C:\Win
acpidev.inf                             C:\Win
acpipagr.inf                            C:\Win
acpipmi.inf                             C:\Win
```

Administrator: Command Prompt

```
C:\Windows\system32>driverquery

Module Name  Display Name           Kernel
============ ====================== ======
1394ohci     1394 OHCI Compliant Ho Kernel
3ware        3ware                  Kernel
ACPI         Microsoft ACPI Driver  Kernel
AcpiDev      ACPI Devices driver    Kernel
acpiex       Microsoft ACPIEx Drive Kernel
acpipagr     ACPI Processor Aggrega Kernel
AcpiPmi      ACPI Power Meter Drive Kernel
acpitime     ACPI Wake Alarm Driver Kernel
AcxHdAudio   ACX HD Audio Driver    Kernel
```

Administrator: Windows PowerShell

```
PS C:\Windows\system32> fltmc

Filter Name                     Num Instances    Altitude    Frame
------------------------------  -------------  ------------  -----
bindflt                             1          409800          0
WdFilter                            9          328010          0
storqosflt                          0          244000          0
wcifs                               0          189900          0
PrjFlt                              0          189800          0
CldFlt                              1          180451          0
FileCrypt                           0          141100          0
luafv                               1          135000          0
npsvctrig                           1           46000          0
Wof                                 3           40700          0
FileInfo                            9           40500          0

PS C:\Windows\system32>
```

# Answer

- Step #2 (attack): malicious kernel ops
  - Know your environment: which drivers are supposed to be loaded? Can you tell the difference?
  - Change your analysis pattern
  - Extracting/Dumping a sample from memory
  - Snapshotting the OS (offline analysis)
  - Grabbing the SYS file on disk (for static analysis)

Situation
Complication
Question
**Answer**

# Answer

Situation
Complication
Question
**Answer**

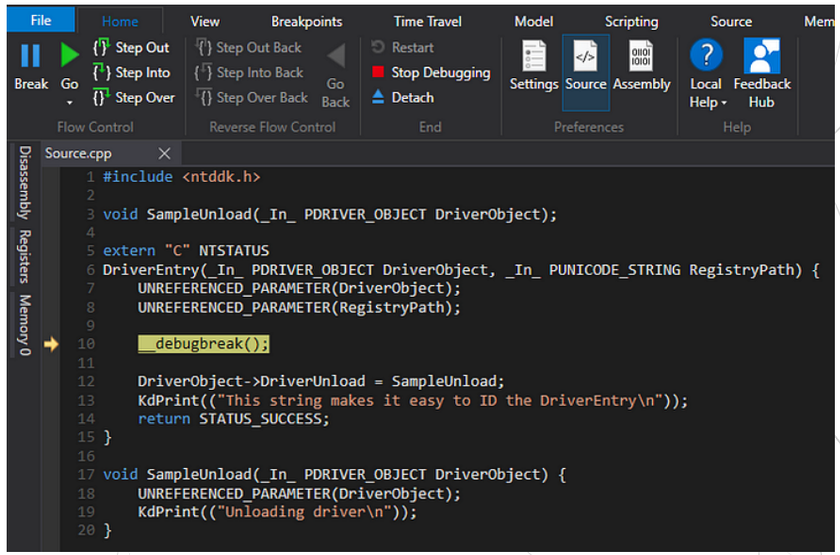
Do you know who to call?


```
.text:000000000001063C
.text:000000000001063C
.text:000000000001063C
.text:000000000001063C                          public start
.text:000000000001063C                          start proc near
.text:000000000001063C
.text:000000000001063C                          var_58= dword ptr -58h
.text:000000000001063C                          var_50= byte ptr -50h
.text:000000000001063C                          var_48= qword ptr -48h
.text:000000000001063C                          var_38= byte ptr -38h
.text:000000000001063C                          var_28= byte ptr -28h
.text:000000000001063C                          arg_10= qword ptr  18h
.text:000000000001063C
.text:000000000001063C 48 53                    push    rbx
.text:000000000001063E 57                       push    rdi
.text:000000000001063F 48 83 EC 68              sub     rsp, 68h
.text:0000000000010643 48 8B D9                 mov     rbx, rcx
.text:0000000000010646 48 8D 3D B3 F9 FF FF     lea     rdi, __ImageBase
.text:000000000001064D 4C 8D 1D 2C 02 00 00     lea     r11, unk_10880
.text:0000000000010654 33 C9                    xor     ecx, ecx
```

```
.text:0000000000010656
.text:0000000000010656                          loc_10656:
.text:0000000000010656 0F B7 84 39 74 07 00 00  movzx   eax, word ptr [rcx+rdi+774h]
.text:000000000001065E 66 42 89 04 19           mov     [rcx+r11], ax
.text:0000000000010663 48 83 C1 02              add     rcx, 2
.text:0000000000010667 66 85 C0                 test    ax, ax
.text:000000000001066A 75 EA                    jnz     short loc_10656
```

```
.text:000000000001066C 48 8D 15 0D 03 00 00     lea     rdx, unk_10980
.text:0000000000010673 49 8B CB                 mov     rcx, r11
.text:0000000000010676 E8 31 FD FF FF           call    sub_103AC
.text:000000000001067B 48 8D 4C 24 40           lea     rcx, [rsp+78h+var_38]
.text:0000000000010680 49 8B D3                 mov     rdx, r11
.text:0000000000010683 FF 15 7F FC FF FF        call    cs:RtlInitUnicodeString
.text:0000000000010689 4C 8D 9C 24 90 00 00 00  lea     r11, [rsp+78h+arg_10]
.text:0000000000010691 4C 8D 44 24 40           lea     r8, [rsp+78h+var_38]
.text:0000000000010696 4C 89 5C 24 30           mov     [rsp+78h+var_48], r11
.text:000000000001069B 41 B9 01 AA 00 00        mov     r9d, 0AA01h
.text:00000000000106A1 33 D2                    xor     edx, edx
.text:00000000000106A3 48 8B CB                 mov     rcx, rbx
.text:00000000000106A6 C6 44 24 28 00           mov     [rsp+78h+var_50], 0
.text:00000000000106AB C7 44 24 20 00 00 00 00  mov     [rsp+78h+var_58], 0
.text:00000000000106B3 FF 15 6F FC FF FF        call    cs:IoCreateDevice
.text:00000000000106B9 85 C0                    test    eax, eax
```

https://medium.com/@0x4ndr3/starting-dynamic-analysis-on-a-windows-x64-rootkit-8c7a74871fda



**"Under pressure, you don't rise to the occasion, you sink to the level of your training." - Navy Seal**

These slides →

# THANK YOU !

https://www.linkedin.com/in/aflima/

Conferences: https://github.com/0x4ndr3/Presentations

https://www.youtube.com/@0x4ndr3

0x4ndr3

Next... →

Red Team (infrastructure and payload development) automation

BSIDES OSLO

21 September 2023