Reason

Webcam Interception and Protection in Kernel Mode in Windows

Michael Maltsev
VB2019 London
October 3, 2019

About Us

Reason

- Founded in 2012 by Andrew Newman, a leading security expert.
- Focused on privacy protection.
- Products:
  - Reason Antivirus (formerly known as Reason Core Security)
  - Should I Remove It?
  - Unchecky
- The Reason Antivirus engine scans over 1B files in 180 countries a day.

Reason Cybersecurity

Outline

- The importance of webcam security
- Multimedia frameworks
  - DShow Bridge
  - Frame Server
- Attack strategies
- Protection driver development
- Existing protection solutions

The Importance of Webcam Security

About Us

The Importance of Webcam Security

Multimedia Frameworks

Attack Strategies

Protection Driver Development

Existing Protection Solutions

The Importance of Webcam Security

Global Surveillance

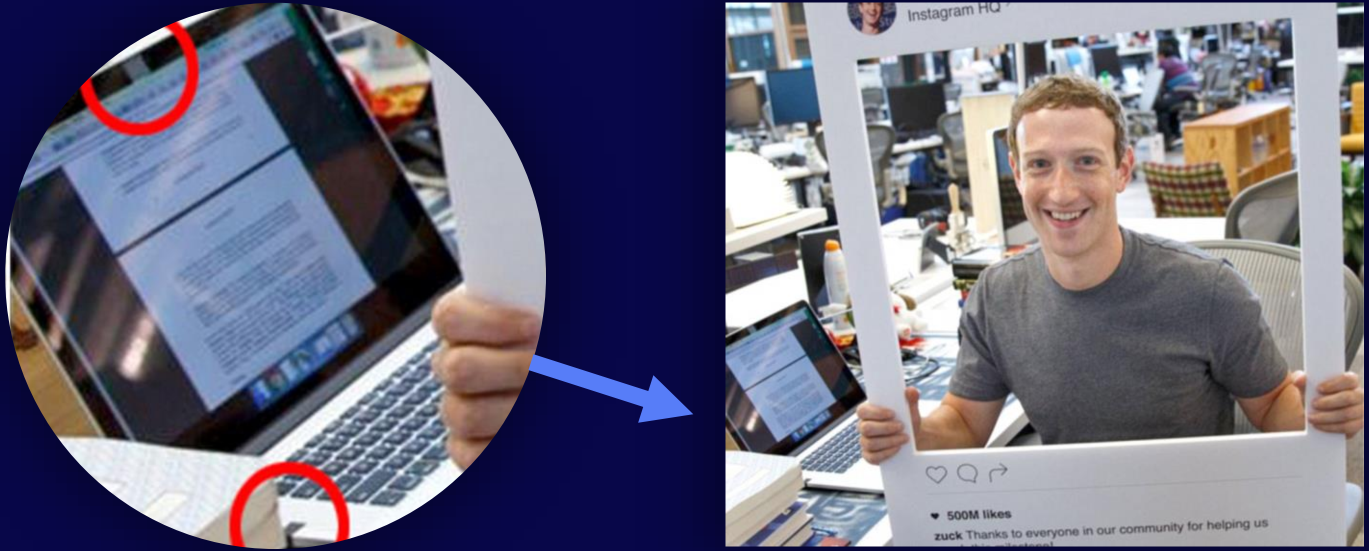
NSA, Hacking Team, etc.

Stalkerware

e.g. FlexiSPY

Webcam Ransom

The Importance of Webcam Security



Reason

The Importance of Webcam Security

"Since being founded in 2003, the Italian spyware vendor Hacking Team gained notoriety for selling surveillance tools to governments and their agencies across the world. […] The capabilities of its flagship product, the Remote Control System (RCS), include […] remotely activating a device's webcam and microphone."

Filip Kafka, WeLiveSecurity (ESET)

"Mark Zuckerberg masks Mac webcam and microphone […] FBI director James Comey has previously said he also covers his laptop's webcam to prevent hackers spying on him."

Kim Zetter, WIRED

"According to The Intercept, the NSA uses a plug-in called GUMFISH to take over cameras on infected machines and snap photos."

BBC News

Multimedia Frameworks

About Us

The Importance of Webcam Security

Multimedia Frameworks

Attack Strategies

Protection Driver Development

Existing Protection Solutions

Multimedia Frameworks

1992
VfW
Obsolete

1996
DirectShow

2006
Media Foundation

Microsoft Windows 95

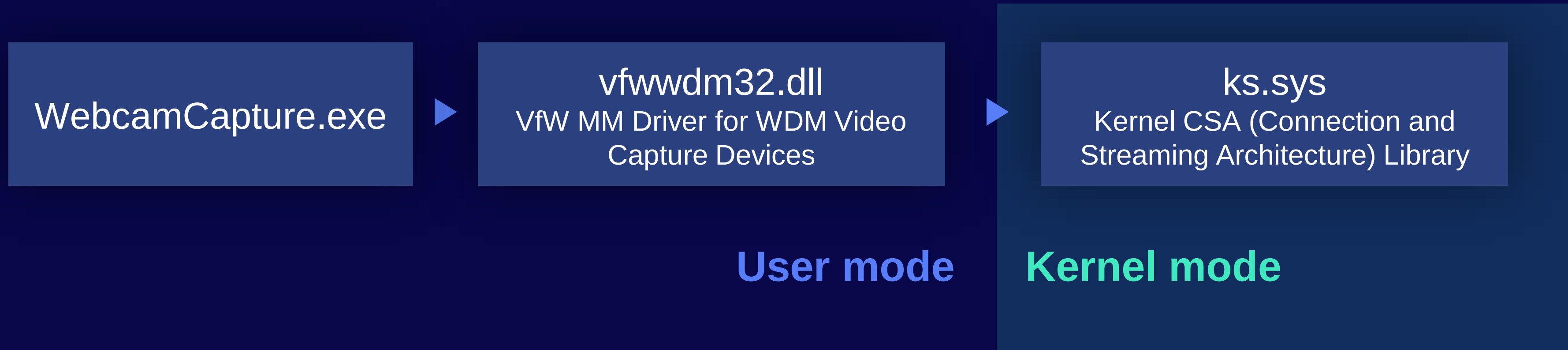
Microsoft Windows 98

Windows Vista

Multimedia Frameworks

VfW (Video for Windows)

- A multimedia framework that enables applications to process video data.
- First introduced in 1992. Became an integral component of Windows 95 and later.
- Replaced by DirectShow in 1996.
- Nowadays implemented by the VFWWDM32 driver as a backward compatibility layer.



WebcamCapture.exe

vfwwdm32.dll
VfW MM Driver for WDM Video Capture Devices

ks.sys
Kernel CSA (Connection and Streaming Architecture) Library

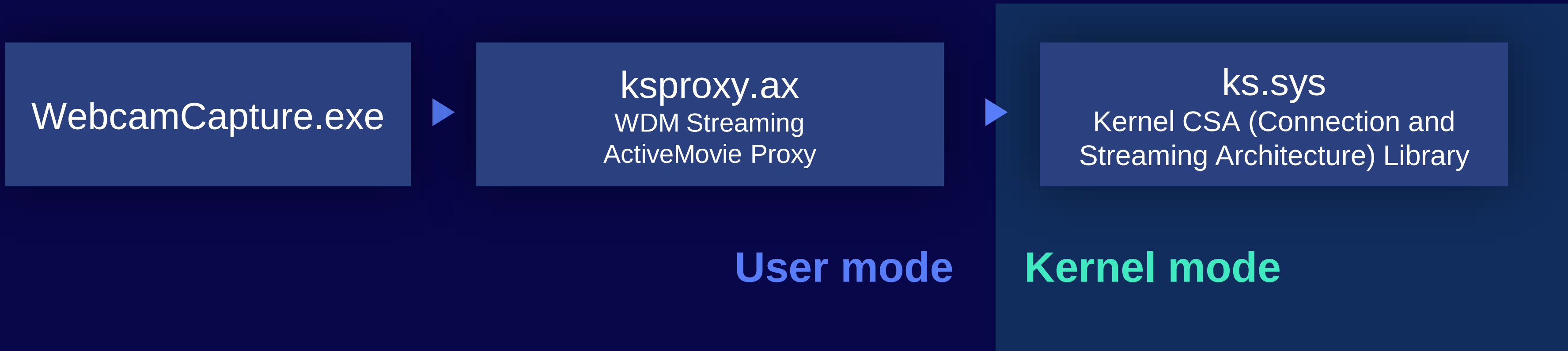
User mode

Kernel mode

Multimedia Frameworks

Direct Show

- An architecture for streaming media based on the Component Object Model (COM).
- Announced in 1996, originally named ActiveMovie.
- Became a standard component of all Windows operating systems starting with Windows 98.
- Probably the most popular API for interacting with the camera on Windows today.



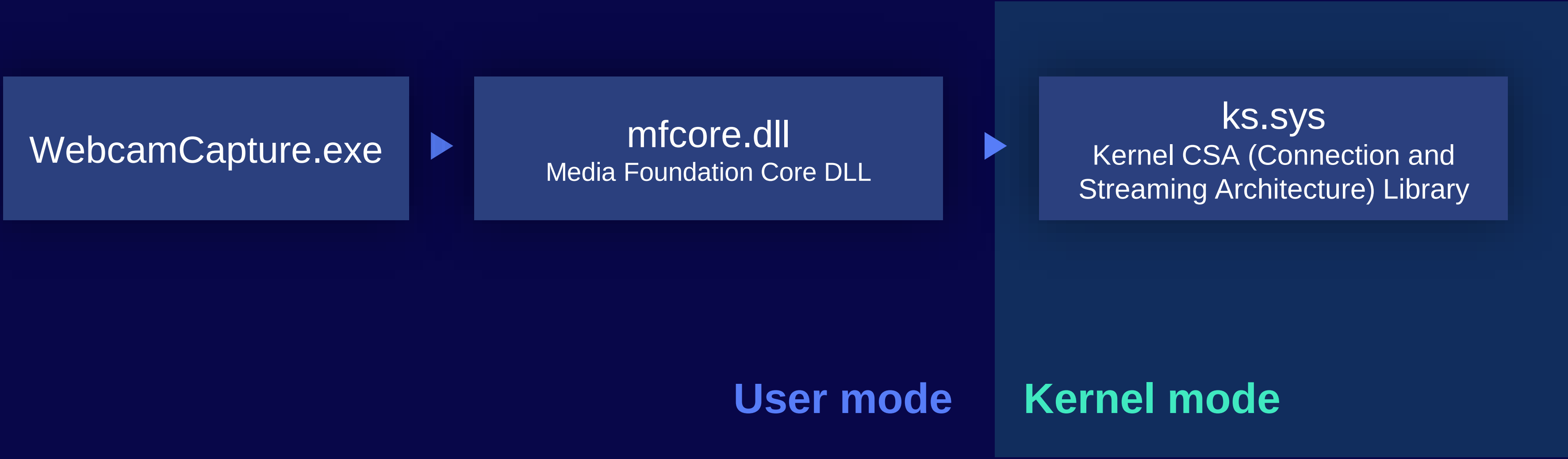
Multimedia Frameworks

Media Foundation

- A multimedia platform based on the Component Object Model (COM), intended to replace DirectShow (but apparently is not there yet).
- Introduced in Windows Vista (2006), enhanced in Windows 7 and further enhanced in Windows 8.
- *“Media Foundation has a couple features that DirectShow did not. The playback experience should be better (fewer glitches), and it also has much more robust support for content protection systems.”*
  Becky Weiss, Microsoft

Multimedia Frameworks

Media Foundation



WebcamCapture.exe

mfcore.dll
Media Foundation Core DLL

ks.sys
Kernel CSA (Connection and Streaming Architecture) Library

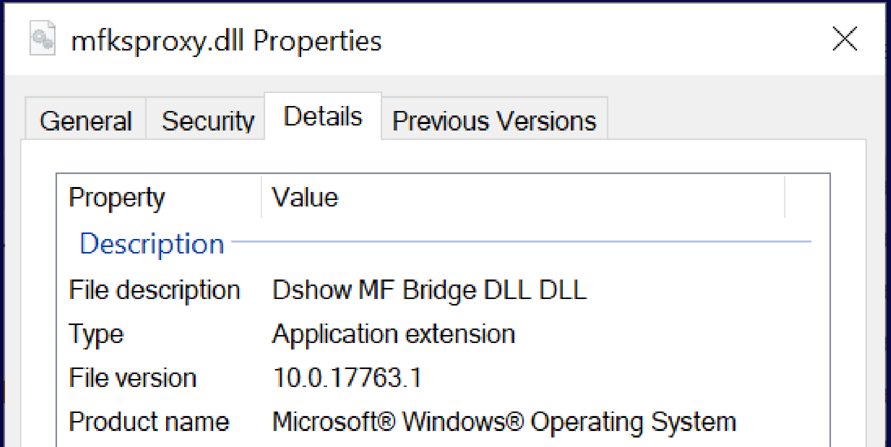
User mode

Kernel mode

Reason

DShow Bridge

- A proxy to allow applications designed for DirectShow to use Media Foundation.
- Introduced in Windows 10 Anniversary Update (Version 1607, August 2016).
- Implemented in mfksproxy.dll.



mfksproxy.dll Properties

Property	Value
Description	
File description	Dshow MF Bridge DLL DLL
Type	Application extension
File version	10.0.17763.1
Product name	Microsoft® Windows® Operating System

DShow Bridge

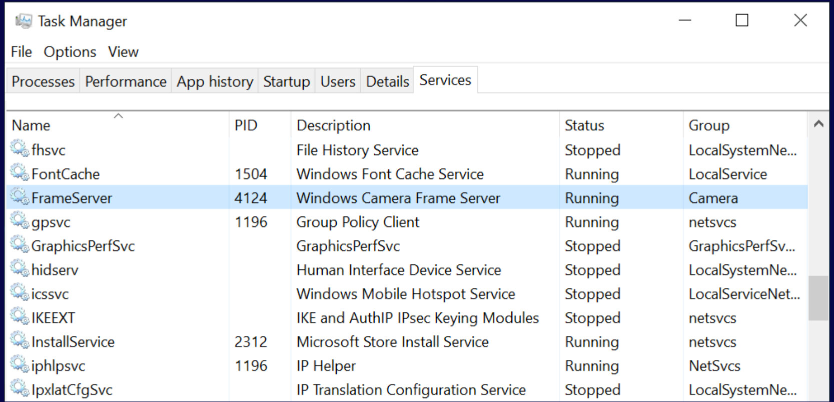
Enabled if:

- DShow Bridge is enabled for the camera device.
  - HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\<DeviceVID&PID>\<DeviceInstance>\Device Parameters
  - Enabled if the **EnableDshowRedirection** DWORD value has the 0x00000001 bit set.
- Capturing application is not blacklisted from using DShow Bridge:
  - HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\OEM\DshowBridge\<number>
  - HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Microsoft\OEM\DshowBridge\<number>
  - There’s an existing entry for “Launcher_Main.exe” from Logitech in the above branches.

Frame Server

- Introduced in Windows 10 Anniversary Update together with DShow Bridge.
- A service virtualizing a camera device, allowing the device to be shared between multiple applications.
- In practice, camera device sharing is reserved for the system.

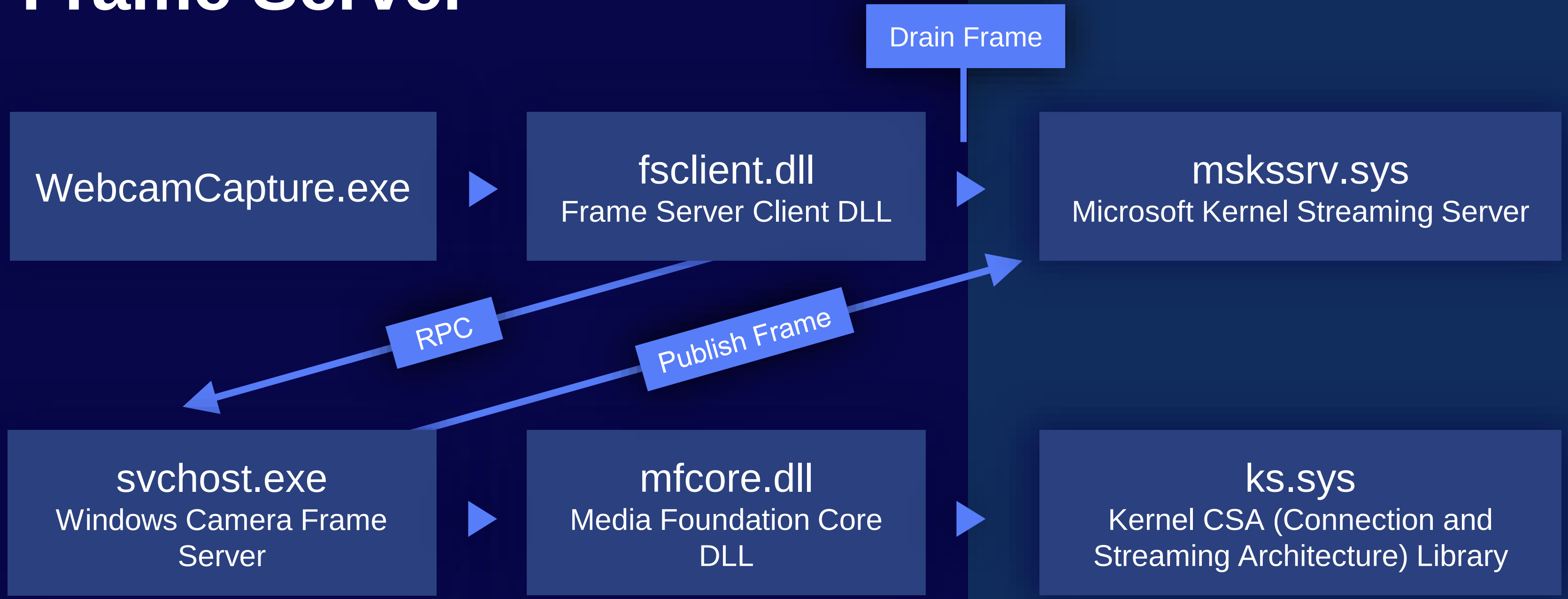
“It was important for us to enable concurrent camera access, so Windows Hello, Microsoft Hololens and other products and features could reliably assume that the camera would be available at any given time”
Mike M, Windows Camera Team


Task Manager
File Options View
Processes Performance App history Startup Users Details Services

Name	PID	Description	Status	Group
fhsvc		File History Service	Stopped	LocalSystemNe...
FontCache	1504	Windows Font Cache Service	Running	LocalService
FrameServer	4124	Windows Camera Frame Server	Running	Camera
gpsvc	1196	Group Policy Client	Running	netsvcs
GraphicsPerfSvc		GraphicsPerfSvc	Stopped	GraphicsPerfSv...
hidserv		Human Interface Device Service	Stopped	LocalSystemNe...
icssvc		Windows Mobile Hotspot Service	Stopped	LocalServiceNet...
IKEEXT		IKE and AuthIP IPsec Keying Modules	Stopped	netsvcs
InstallService	2312	Microsoft Store Install Service	Running	netsvcs
iphlpsvc	1196	IP Helper	Running	NetSvcs
IpxlatCfgSvc		IP Translation Configuration Service	Stopped	LocalSystemNe...

Reason

Frame Server



Drain Frame

WebcamCapture.exe

fsclient.dll
Frame Server Client DLL

mskssrv.sys
Microsoft Kernel Streaming Server

RPC

Publish Frame

svchost.exe
Windows Camera Frame Server

mfcore.dll
Media Foundation Core DLL

ks.sys
Kernel CSA (Connection and Streaming Architecture) Library

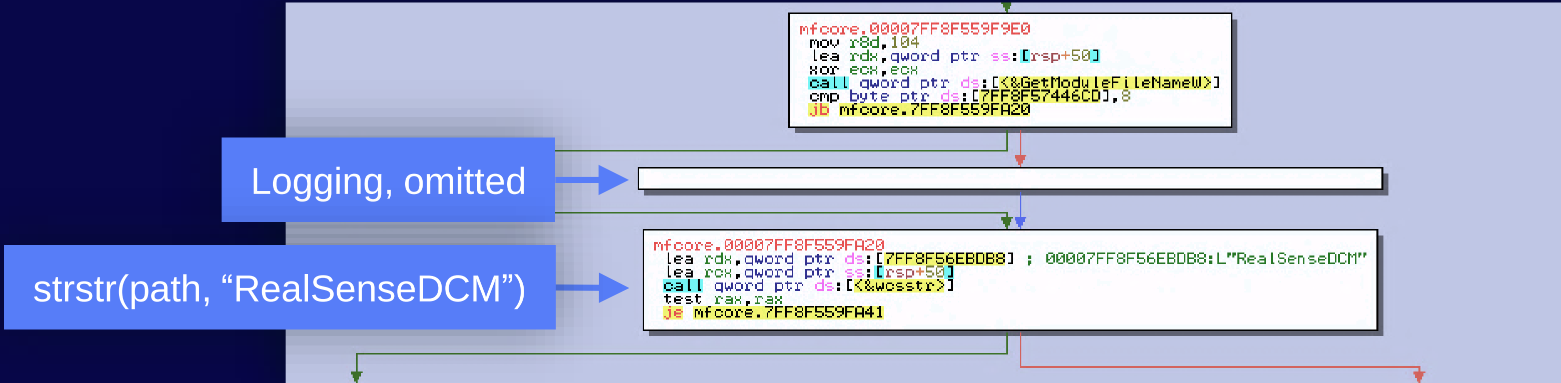
User mode

Kernel mode

Frame Server

Enabled if:

- Application is using Media Foundation (or DirectShow with DShow Bridge enabled).
- Frame Server is not disabled in the registry.
    - The EnableFrameServerMode DWORD value in the following registry branch: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Media Foundation\Platform
    - Enabled by default if the value is absent.
- The application path doesn’t contain “RealSenseDCM”.



```
mfcore.00007FF8F559F9E0
 mov r8d,104
 lea rdx,qword ptr ss:[rsp+50]
 xor ecx,ecx
 call qword ptr ds:[<&GetModuleFileNameW>]
 cmp byte ptr ds:[7FF8F57446CD],8
 jb mfcore.7FF8F559FA20
```

Logging, omitted

```
mfcore.00007FF8F559FA20
 lea rdx,qword ptr ds:[7FF8F56EBDB8] ; 00007FF8F56EBDB8:L"RealSenseDCM"
 lea rcx,qword ptr ss:[rsp+50]
 call qword ptr ds:[<&wcsstr>]
 test rax,rax
 je mfcore.7FF8F559FA41
```

strstr(path, “RealSenseDCM”)

Attack Strategies

About Us

The Importance of Webcam Security

Multimedia Frameworks

Attack Strategies

Protection Driver Development

Existing Protection Solutions

Reasōn

Attack Strategies

Simple

Just record

Stealth

Use an existing session

God Mode

Just record, LED off

Attack Strategies (1/3)

Simple

- Just start recording.
- Advantages:
  - Just works, no questions are asked by Windows.
    Note: Since Windows 10 Fall Creators Update, the user is prompted for access to the camera device, but it only applies to Store apps.
  - Many code examples and available programs that can be used.
- Disadvantages:
  - Camera usage will turn on the indicator LED, giving the victim an indication that somebody is watching him.

Attack Strategies (2/3)

Stealth

- Use an existing recording session. For example, inject into Skype and intercept the camera traffic.
- Advantages:
    - No unusual indicator LED activity, more difficult for the victim to detect.
- Disadvantages:
    - Not as easy to implement.
    - The attacker can’t choose when to record. The recording sessions are chosen by the victim, and might be less valuable.
    - Can be intrusive depending on the implementation, increases the chances of being detected by a security software.

Attack Strategies (3/3)

God Mode

- Like the first option, but keep the indicator LED turned off.
- Ideally, it shouldn’t be possible to use the camera without turning on the indicator LED. In practice, some models provide this option, or have a design flaw that allows it.
- Examples:
  - *“I've found info on Logitech Webcams, where you can turn off the LED in the registry keys.”*
    psalomonsen, Information Security Stack Exchange
  - *“[…] on our Dell laptop, we find the DLL that comes with the RealTek drivers for our webcam. We quickly zero in on the exported function "TurnOnOffLED()".“*
    Robert Graham, Errata Security: How to disable webcam light on Windows

Attack Strategies (3/3)

God Mode

- Advantages:
  - The attacker can record at any time without a visible indication.
- Disadvantages:
  - Device-specific, hard or impossible to do on well-designed devices.

Protection Driver Development

About Us

The Importance of Webcam Security

Multimedia Frameworks

Attack Strategies

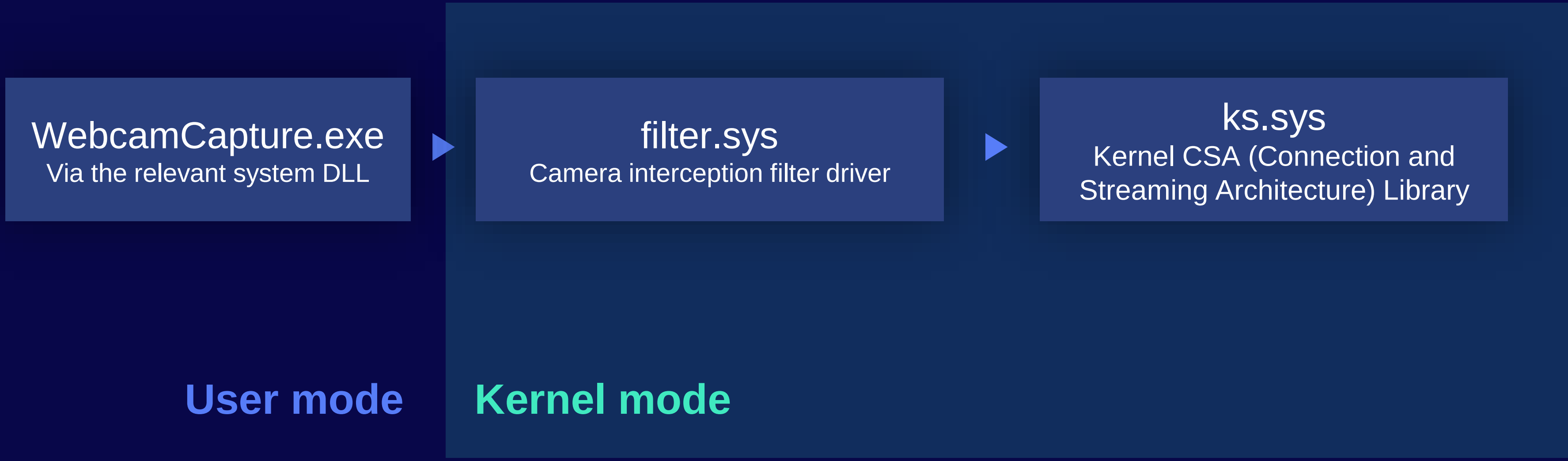
Protection Driver Development

Existing Protection Solutions

Protection Driver Development

- Mostly undocumented, not much information on the internet.
- After some research we can come to the conclusion that the way to go is to implement a filter driver.
- Let’s implement a filter driver which logs the IRPs (I/O request packets) that pass through.

Protection Driver Development



WebcamCapture.exe
Via the relevant system DLL

filter.sys
Camera interception filter driver

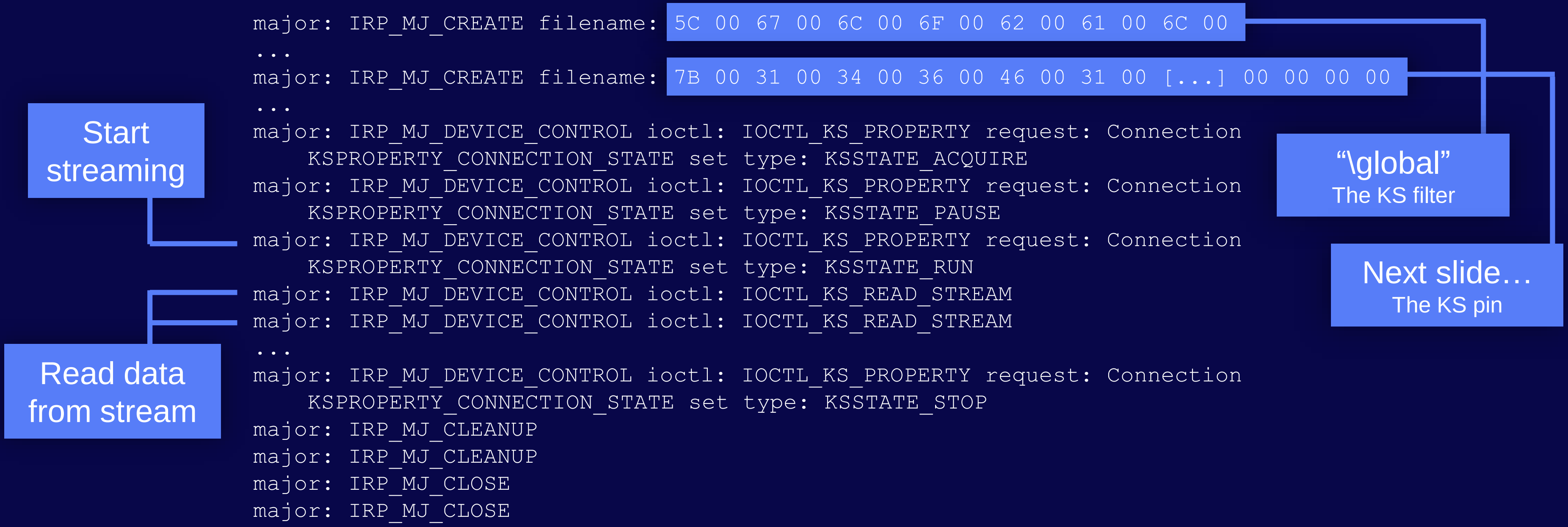
ks.sys
Kernel CSA (Connection and Streaming Architecture) Library

User mode

Kernel mode

Protection Driver Development

The Interesting IRPs



```
major: IRP_MJ_CREATE filename: 5C 00 67 00 6C 00 6F 00 62 00 61 00 6C 00
...
major: IRP_MJ_CREATE filename: 7B 00 31 00 34 00 36 00 46 00 31 00 [...] 00 00 00 00
...
major: IRP_MJ_DEVICE_CONTROL ioctl: IOCTL_KS_PROPERTY request: Connection
    KSPROPERTY_CONNECTION_STATE set type: KSSTATE_ACQUIRE
major: IRP_MJ_DEVICE_CONTROL ioctl: IOCTL_KS_PROPERTY request: Connection
    KSPROPERTY_CONNECTION_STATE set type: KSSTATE_PAUSE
major: IRP_MJ_DEVICE_CONTROL ioctl: IOCTL_KS_PROPERTY request: Connection
    KSPROPERTY_CONNECTION_STATE set type: KSSTATE_RUN
major: IRP_MJ_DEVICE_CONTROL ioctl: IOCTL_KS_READ_STREAM
major: IRP_MJ_DEVICE_CONTROL ioctl: IOCTL_KS_READ_STREAM
...
major: IRP_MJ_DEVICE_CONTROL ioctl: IOCTL_KS_PROPERTY request: Connection
    KSPROPERTY_CONNECTION_STATE set type: KSSTATE_STOP
major: IRP_MJ_CLEANUP
major: IRP_MJ_CLEANUP
major: IRP_MJ_CLOSE
major: IRP_MJ_CLOSE
```

Start streaming

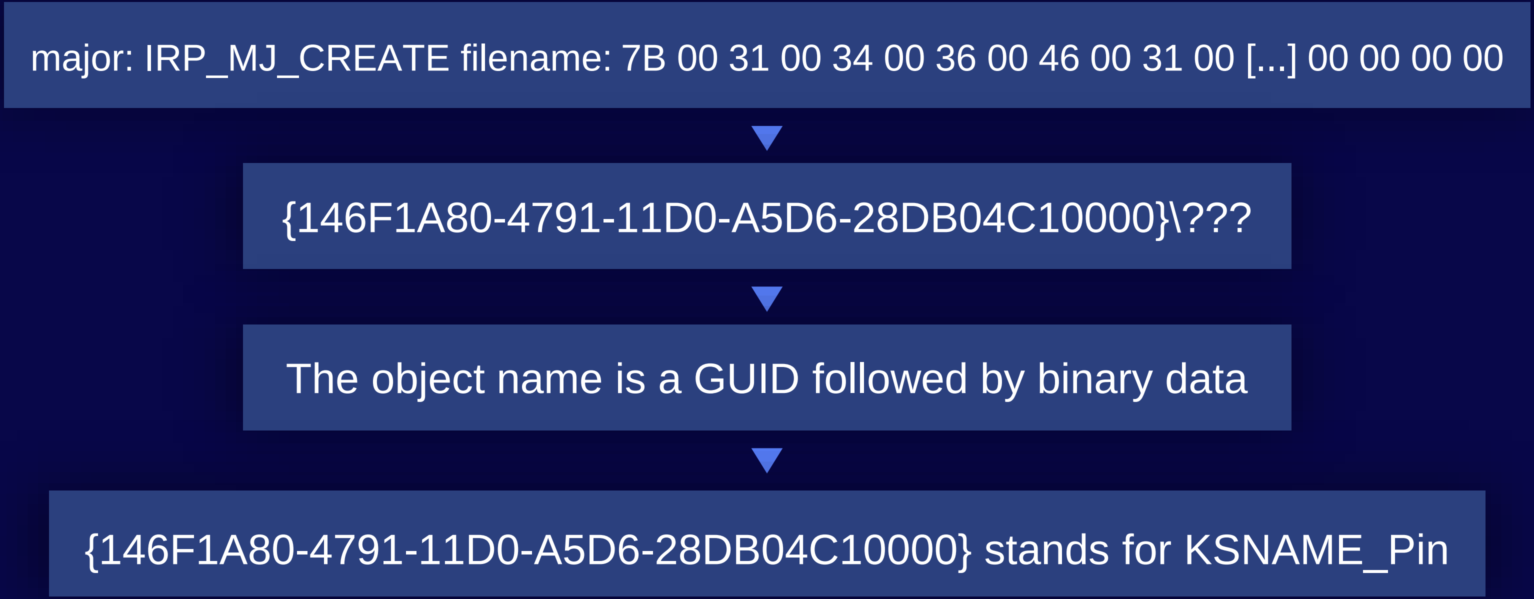
Read data from stream

“\global”
The KS filter

Next slide…
The KS pin

Protection Driver Development

The IRP_MJ_CREATE for the KS Pin



major: IRP_MJ_CREATE filename: 7B 00 31 00 34 00 36 00 46 00 31 00 [...] 00 00 00 00

{146F1A80-4791-11D0-A5D6-28DB04C10000}\???

The object name is a GUID followed by binary data

{146F1A80-4791-11D0-A5D6-28DB04C10000} stands for KSNAME_Pin

Reason

Protection Driver Development

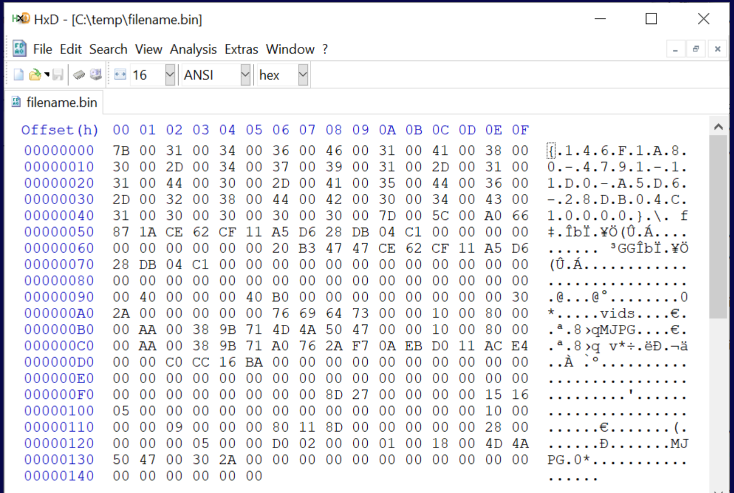
The IRP_MJ_CREATE for the KS Pin

Format:

{146F1A80-4791…}\<KSPIN_CONNECT><KSDATAFORMAT>

The **KSPIN_CONNECT** structure describes the connection details, such as the pin ID.

The **KSDATAFORMAT** structure describes the image format.


HxD - [C:\temp\filename.bin]
File Edit Search View Analysis Extras Window ?
16
ANSI
hex
filename.bin

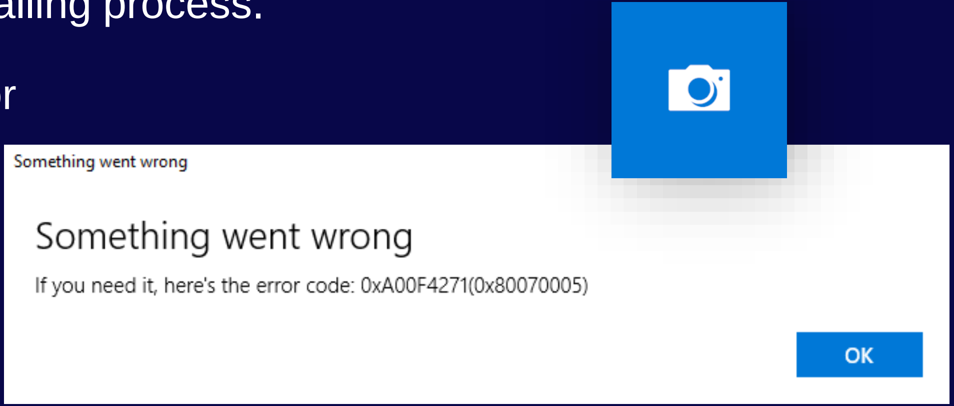
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  7B 00 31 00 34 00 36 00 46 00 31 00 41 00 38 00  {.1.4.6.F.1.A.8.
00000010  30 00 2D 00 34 00 37 00 39 00 31 00 2D 00 31 00  0.-.4.7.9.1.-.1.
00000020  31 00 44 00 30 00 2D 00 41 00 35 00 44 00 36 00  1.D.0.-.A.5.D.6.
00000030  2D 00 32 00 38 00 44 00 42 00 30 00 34 00 43 00  -.2.8.D.B.0.4.C.
00000040  31 00 30 00 30 00 30 00 30 00 7D 00 5C 00 A0 66  1.0.0.0.0.}.\. f
00000050  87 1A CE 62 CF 11 A5 D6 28 DB 04 C1 00 00 00 00  ‡.ÎbÏ.¥Ö(Û.Á....
00000060  00 00 00 00 00 00 20 B3 47 47 CE 62 CF 11 A5 D6  ...... ³GGÎbÏ.¥Ö
00000070  28 DB 04 C1 00 00 00 00 00 00 00 00 00 00 00 00  (Û.Á............
00000080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000090  00 40 00 00 00 40 B0 00 00 00 00 00 00 00 00 30  .@...@°........0
000000A0  2A 00 00 00 00 00 76 69 64 73 00 00 10 00 80 00  *.....vids....€.
000000B0  00 AA 00 38 9B 71 4D 4A 50 47 00 00 10 00 80 00  .ª.8›qMJPG....€.
000000C0  00 AA 00 38 9B 71 A0 76 2A F7 0A EB D0 11 AC E4  .ª.8›q v*÷.ëÐ.¬ä
000000D0  00 00 C0 CC 16 BA 00 00 00 00 00 00 00 00 00 00  ..À .°..........
000000E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000F0  00 00 00 00 00 00 00 00 8D 27 00 00 00 00 15 16  .........'......
00000100  05 00 00 00 00 00 00 00 00 00 00 00 00 00 10 00  ................
00000110  00 00 09 00 00 00 80 11 8D 00 00 00 00 00 28 00  ......€.......(.
00000120  00 00 00 05 00 00 D0 02 00 00 01 00 18 00 4D 4A  ......Ð.......MJ
00000130  50 47 00 30 2A 00 00 00 00 00 00 00 00 00 00 00  PG.0*...........
00000140  00 00 00 00 00 00                                ......
```

Protection Driver Development

Blocking Access to the Camera

- The simplest solution is to block **IOCTL_KS_PROPERTY** when the **KSSTATE_ACQUIRE** command is sent.
- Note: The kernel streaming IOCTLs use the 'neither' buffering method. Data must be accessed from the context of the calling process.
- The capturing program will get an error while trying to initiate the capture.



Something went wrong

Something went wrong

If you need it, here's the error code: 0xA00F4271(0x80070005)

What if we want to return a fake image instead?

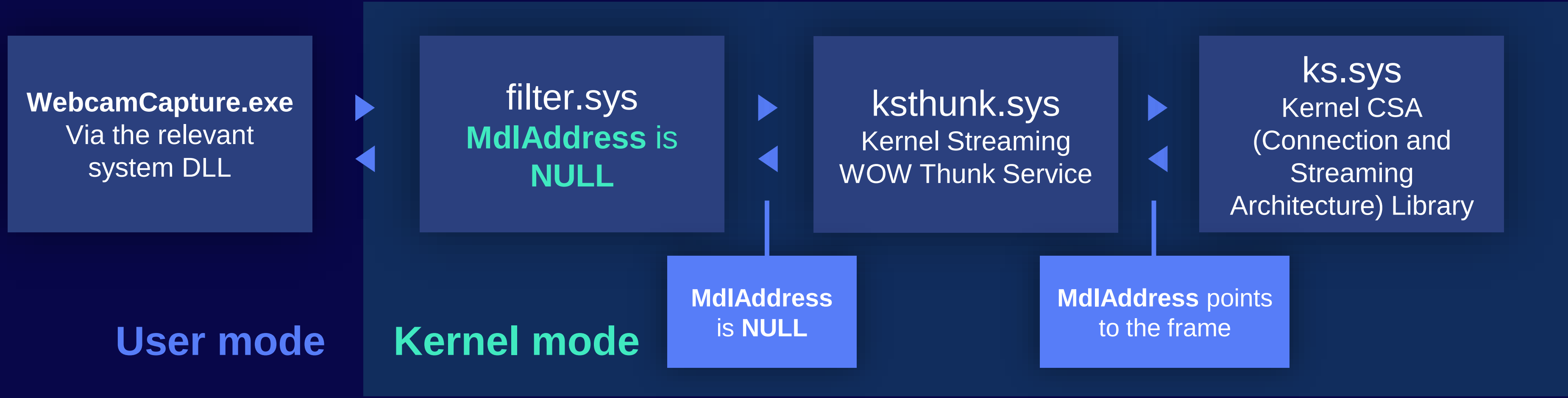
Protection Driver Development

Replacing the Image Frames

- We can post-process **IOCTL_KS_READ_STREAM** and replace the frame image before it leaves the kernel mode.
- The frame image buffer is mapped for us by the camera function driver, and can be accessed via **Irp->MdlAddress**.
- **Irp->MdlAddress** can be **NULL** sometimes. The reason - the **ksthunk** filter driver, designed to provide streaming compatibility for 32-bit programs on a 64-bit system.

Protection Driver Development

Replacing the Image Frames



WebcamCapture.exe
Via the relevant system DLL

filter.sys
MdlAddress is **NULL**

ksthunk.sys
Kernel Streaming WOW Thunk Service

ks.sys
Kernel CSA (Connection and Streaming Architecture) Library

MdlAddress is **NULL**

MdlAddress points to the frame

User mode

Kernel mode

Protection Driver Development

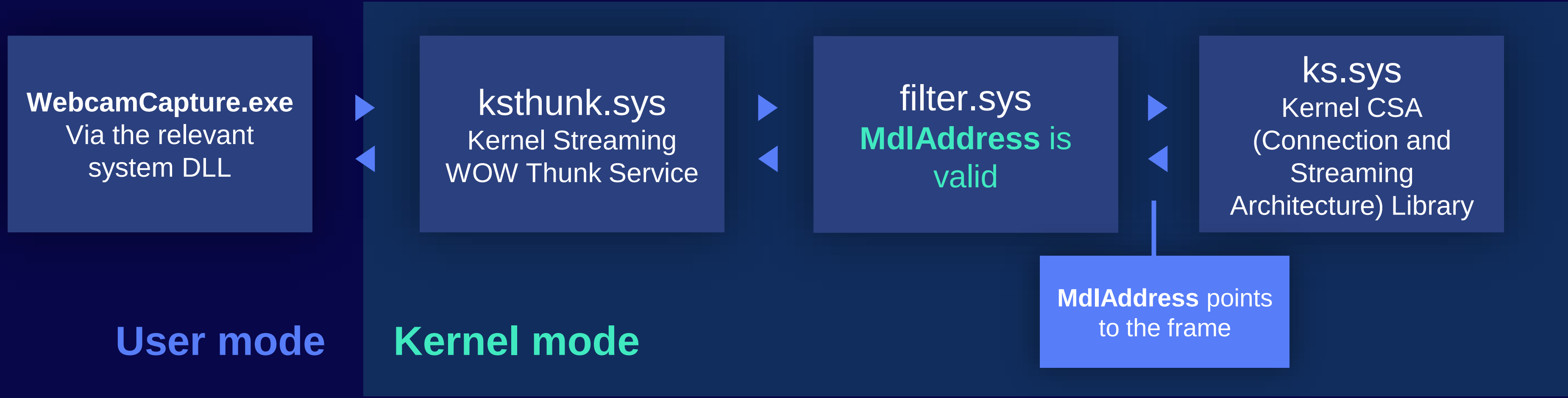
Replacing the Image Frames

- The solution - place our filter driver before **ksthunk** on the stack.

Reason

Protection Driver Development

Replacing the Image Frames


WebcamCapture.exe
Via the relevant system DLL
ksthunk.sys
Kernel Streaming WOW Thunk Service
filter.sys
MdlAddress is valid
ks.sys
Kernel CSA (Connection and Streaming Architecture) Library
MdlAddress points to the frame
User mode
Kernel mode

Reason

Protection Driver Development

Per-process Selective Blocking

- Typical design - maintain a list of allowed programs such as Skype and the web browser while blocking other, unknown programs.
- Used to be straightforward before the introduction of the Frame Server - The process accessing the camera device is the one to be checked against the list.
- Since the introduction of the Frame Server, **svchost.exe** (hosting the Frame Server) will be accessing the camera device every time Media Framework is used.

Protection Driver Development

Per-process Selective Blocking

- We need an alternative way to determine which process is accessing the camera.
- A possible solution - intercept access to **mskssrv.sys**, the Kernel Streaming Server.

Protection Driver Development

Per-process Selective Blocking

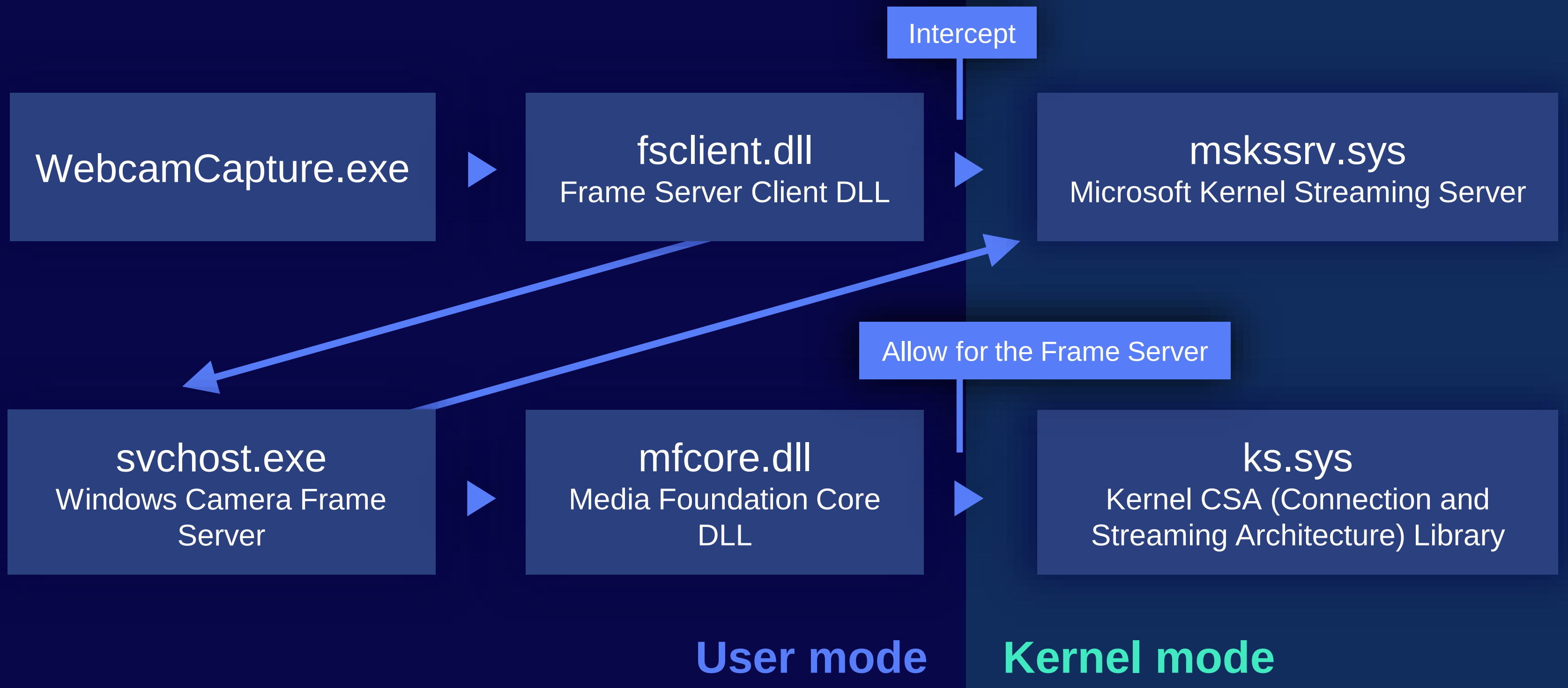
Things to note:

- If intercepting access to **mskssrv.sys**, the process hosting Frame Server needs to be whitelisted.
- You need to be careful about trusting an allowed program, e.g. in case of a code injection scenario.

Reason

Protection Driver Development

Per-process Selective Blocking


Intercept
WebcamCapture.exe
fsclient.dll
Frame Server Client DLL
mskssrv.sys
Microsoft Kernel Streaming Server
Allow for the Frame Server
svchost.exe
Windows Camera Frame Server
mfcore.dll
Media Foundation Core DLL
ks.sys
Kernel CSA (Connection and Streaming Architecture) Library
User mode
Kernel mode

Existing Protection Solutions

About Us

The Importance of Webcam Security

Multimedia Frameworks

Attack Strategies

Protection Driver Development

Existing Protection Solutions

Reason

Existing Protection Solutions

Protection Driver Development

We checked 6 well known security products with camera protection functionality.

	Frame Server support	Upper filter registered (*)	Default action on first run
Product 1	✓	✓	Block and notify
Product 2	✗	✗	Block and notify if untrusted
Product 3	✓	Only for the Frame Server	Always notify, block if untrusted
Product 4	✓	✗	Block and notify if untrusted
Product 5	✗	✓	Block and notify if untrusted
Product 6	✗	✗	Notify, no blocking functionality

(*) If not, an interception device is being attached to the camera driver stack at runtime.

Existing Protection Solutions

Product 1, Bypass Attempt 1

About Us

The Importance of Webcam Security

Multimedia Frameworks

Attack Strategies

Protection Driver Development

Existing Protection Solutions

Reason

Existing Protection Solutions

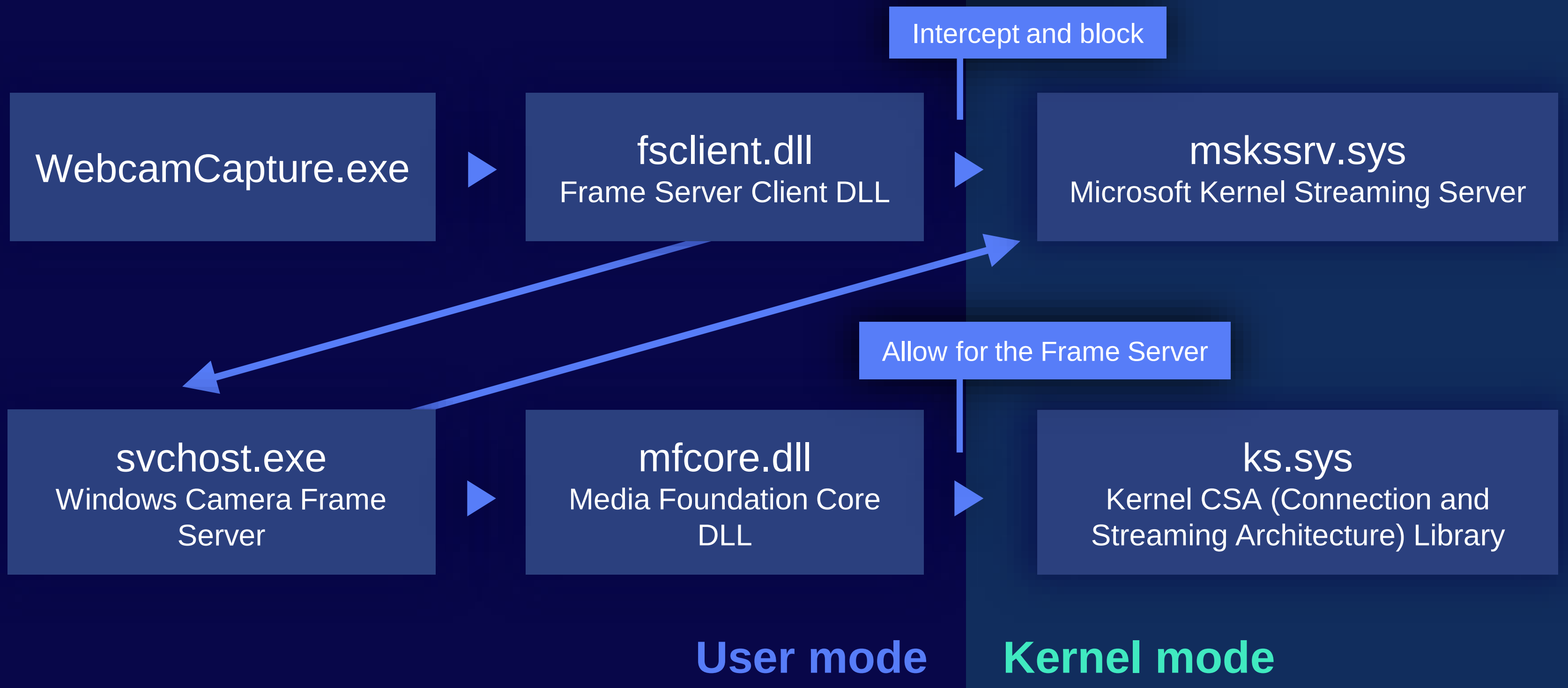
Product 1, Bypass Attempt 1

- After some experimentation we can see that Frame Server is supported by intercepting and blocking access to **mskssrv.sys**, the Kernel Streaming Server.
- In that case, the Frame Server user mode process (**svchost.exe**, hosting the Frame Server) is probably whitelisted.
- What if we inject into it and try to capture?

	Frame Server support	Upper filter registered	Default action on first run
Product 1	✓	✓	Block and notify

Existing Protection Solutions

Product 1, Bypass Attempt 1

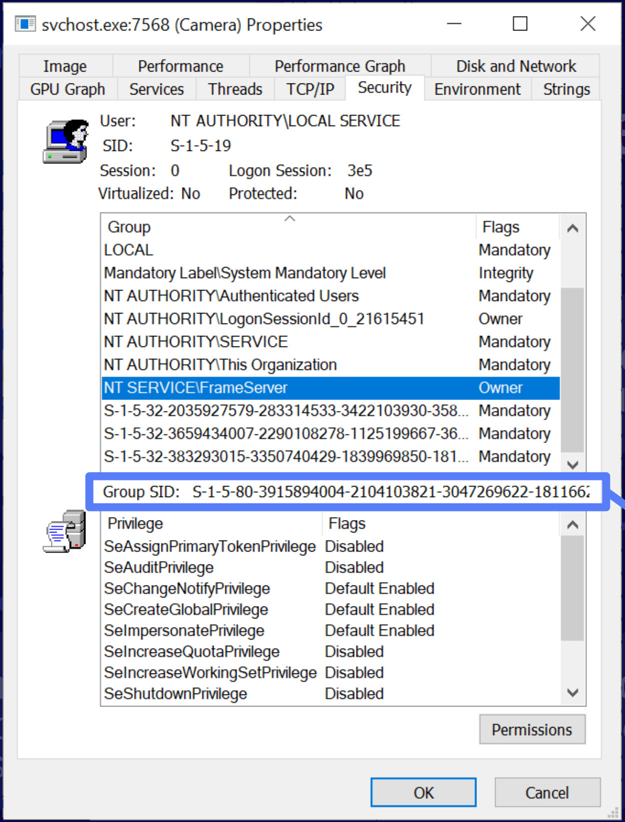


Intercept and block
WebcamCapture.exe
fsclient.dll
Frame Server Client DLL
mskssrv.sys
Microsoft Kernel Streaming Server
Allow for the Frame Server
svchost.exe
Windows Camera Frame Server
mfcore.dll
Media Foundation Core DLL
ks.sys
Kernel CSA (Connection and Streaming Architecture) Library
User mode
Kernel mode

Reasōn

Existing Pr

Prod empt 1



User: NT AUTHORITY\LOCAL SERVICE
SID: S-1-5-19
Session: 0 Logon Session: 3e5
Virtualized: No Protected: No

Group	Flags
LOCAL	Mandatory
Mandatory Label\System Mandatory Level	Integrity
NT AUTHORITY\Authenticated Users	Mandatory
NT AUTHORITY\LogonSessionId_0_21615451	Owner
NT AUTHORITY\SERVICE	Mandatory
NT AUTHORITY\This Organization	Mandatory
NT SERVICE\FrameServer	Owner
S-1-5-32-2035927579-283314533-3422103930-358...	Mandatory
S-1-5-32-3659434007-2290108278-1125199667-36...	Mandatory
S-1-5-32-383293015-3350740429-1839969850-181...	Mandatory

Group SID: S-1-5-80-3915894004-2104103821-3047269622-181166

Privilege	Flags
SeAssignPrimaryTokenPrivilege	Disabled
SeAuditPrivilege	Disabled
SeChangeNotifyPrivilege	Default Enabled
SeCreateGlobalPrivilege	Default Enabled
SeImpersonatePrivilege	Default Enabled
SeIncreaseQuotaPrivilege	Disabled
SeIncreaseWorkingSetPrivilege	Disabled
SeShutdownPrivilege	Disabled

Intercept and block

Webcam

dll

lient DLL

mskssrv.sys

Microsoft Kernel Streaming Server

Allow for the Frame Server

svch

Windows C

S

dll

on Core

ks.sys

Kernel CSA (Connection and Streaming Architecture) Library

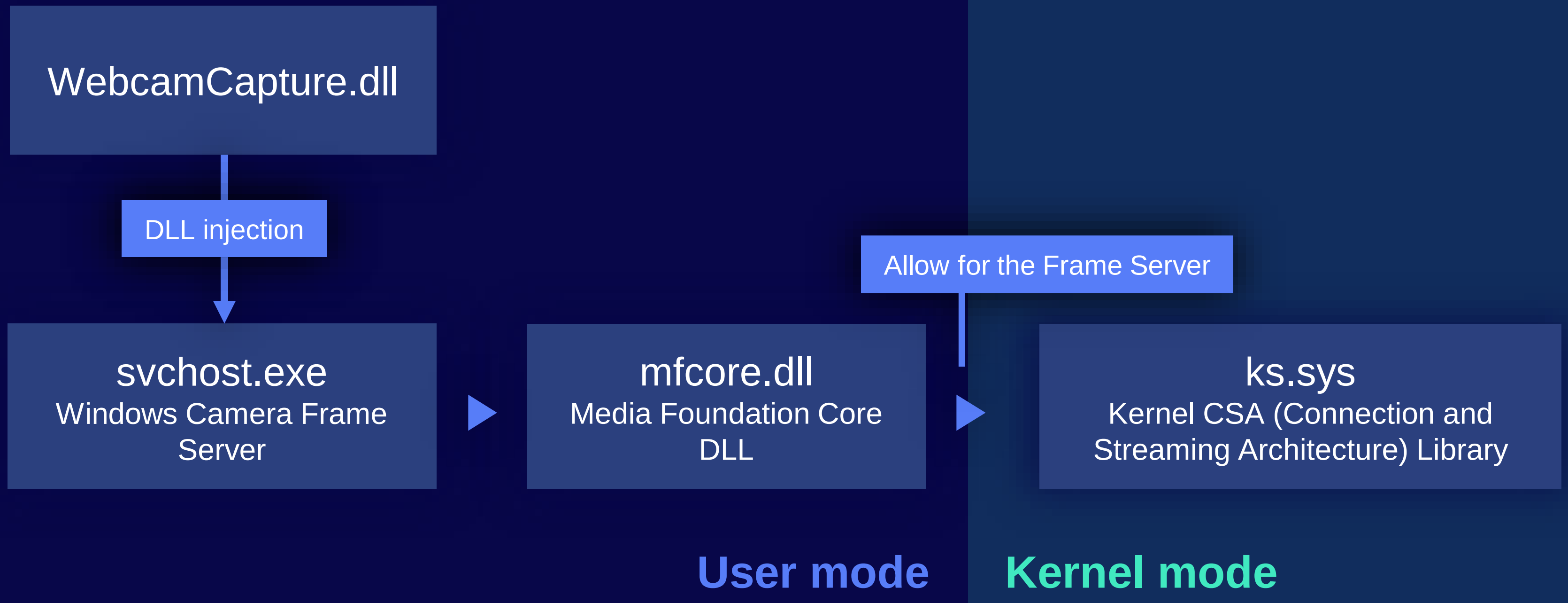
User mode

Kernel mode

Reason

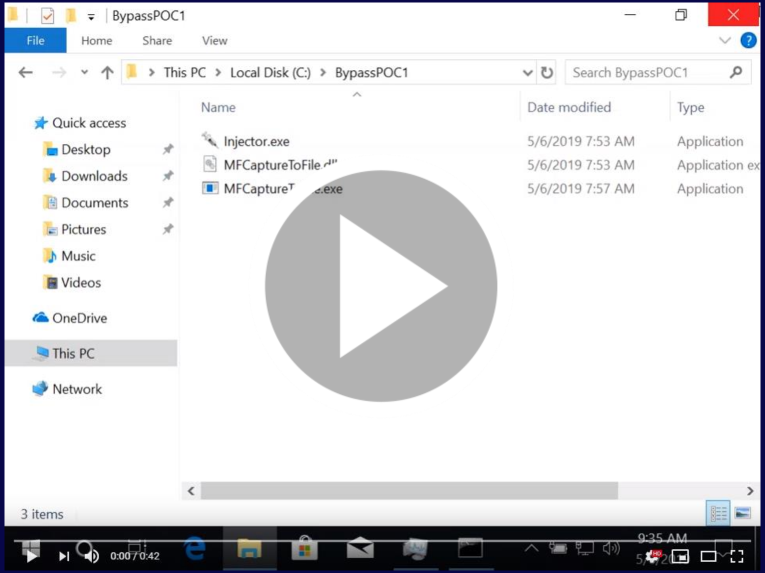
Existing Protection Solutions

Product 1, Bypass Attempt 1


WebcamCapture.dll
DLL injection
svchost.exe
Windows Camera Frame Server
mfcore.dll
Media Foundation Core DLL
Allow for the Frame Server
ks.sys
Kernel CSA (Connection and Streaming Architecture) Library
User mode
Kernel mode

Existing Protection Solutions

Product 1, Bypass Attempt 1



Name	Date modified	Type
Injector.exe	5/6/2019 7:53 AM	Application
MFCaptureToFile.d"	5/6/2019 7:53 AM	Application ex
MFCaptureT[illegible]e.exe	5/6/2019 7:57 AM	Application

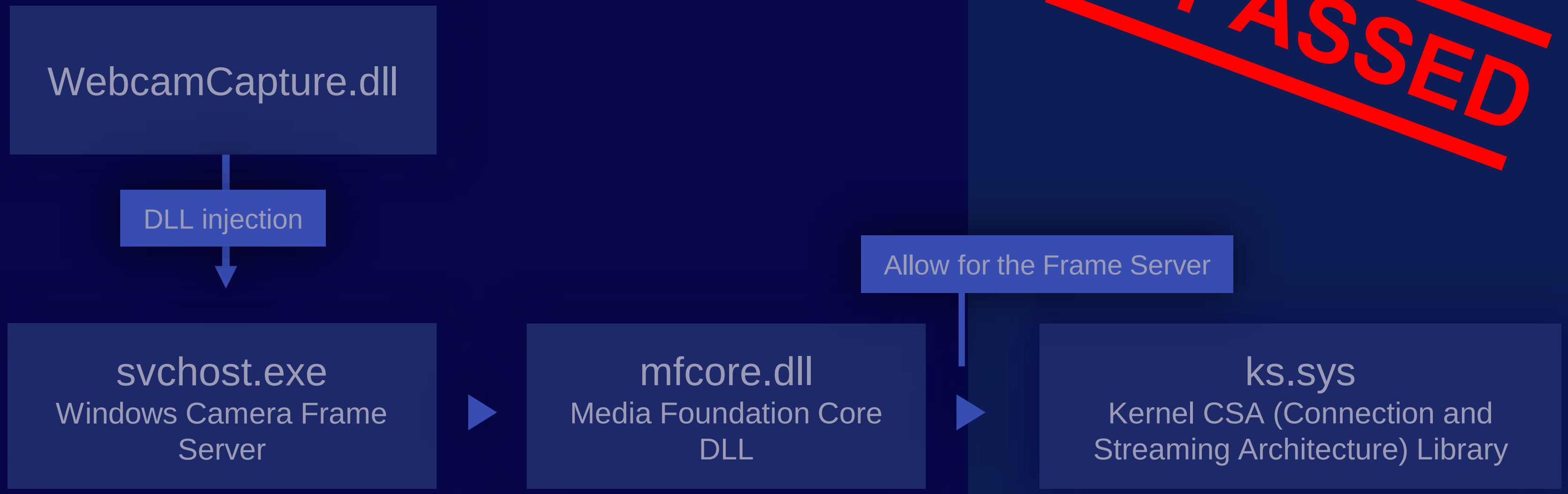
https://youtube.com/watch?v=-bnpcIzlXsA

Reason

Existing Protection Solutions

Product 1, Bypass Attempt 1

BYPASSED



WebcamCapture.dll

DLL injection

svchost.exe
Windows Camera Frame Server

mfcore.dll
Media Foundation Core DLL

Allow for the Frame Server

ks.sys
Kernel CSA (Connection and Streaming Architecture) Library

User mode

Kernel mode

Existing Protection Solutions

Product 1, Bypass Attempt 2

Reasōn

Existing Protection Solutions

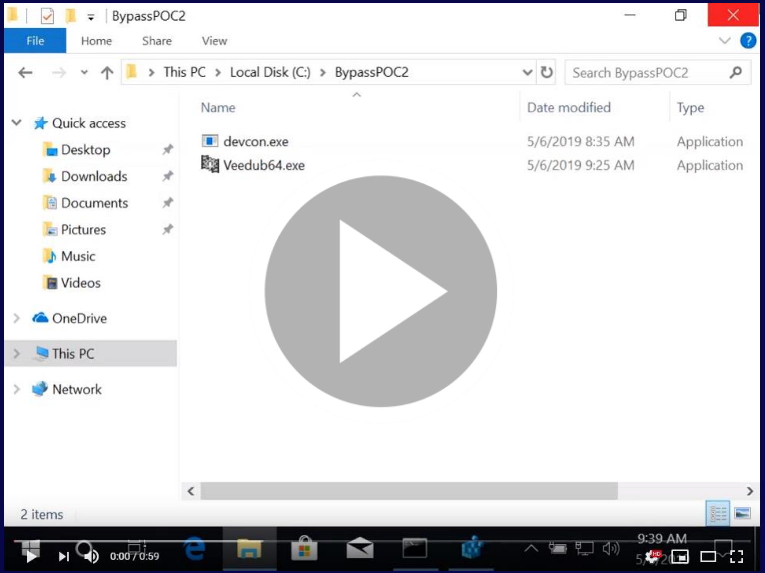
Product 1, Bypass Attempt 2

- Now let’s try something simpler: What happens if we try to remove the registered upper filter driver from the registry?
- Surprisingly, nothing stops us from doing it.
- Now it’s enough to disable and re-enable the camera device to get the security filter driver removed from the driver stack.

	Frame Server support	Upper filter registered	Default action on first run
Product 1	✓	✓	Block and notify

Existing Protection Solutions

Product 1, Bypass Attempt 2



Name	Date modified	Type
devcon.exe	5/6/2019 8:35 AM	Application
Veedub64.exe	5/6/2019 9:25 AM	Application

https://youtube.com/watch?v=JKzoqGk3vTk

Existing Protection Solutions

Product 2

Reason

Existing Protection Solutions

Product 2

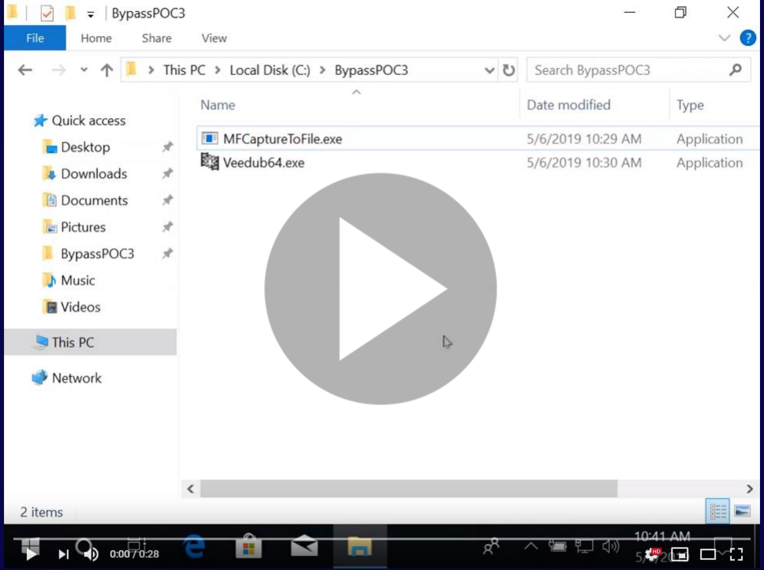
- No Frame Server support - every program accessing the camera via the Frame Server is being reported as **svchost.exe**.
- **svchost.exe**, being a system process, is trusted by default.
- It’s enough to have our capturing program use Media Foundation to bypass the protection.

	Frame Server support	Upper filter registered	Default action on first run
Product 2	✗	✗	Block and notify if untrusted

Reason

Existing Protection Solutions

Product 2



Name	Date modified	Type
MFCaptureToFile.exe	5/6/2019 10:29 AM	Application
Veedub64.exe	5/6/2019 10:30 AM	Application

https://youtube.com/watch?v=Z-64HGY8HtM

Existing Protection Solutions

Product 2

- No Frame Server support - every program accessing the camera via the Frame Server is being reported as **svchost.exe**.

BYPASSED

- **svchost.exe**, being a system process, is trusted by default.
- It’s enough to have our capturing program use Media Foundation to bypass the protection.

	Frame Server support	Upper filter registered	Default action on first run
Product 2	✗	✗	Block and notify if untrusted

Reason

Existing Protection Solutions

Summary

	Frame Server injection	Media Foundation usage	Upper filter removal
Product 1	Bypassed		Bypassed
Product 2		Bypassed	
Product 3	Bypassed		Bypassed
Product 4	Bypassed (*)		
Product 5		Bypassed	Bypassed
Product 6			

(*) Frame Server injection was blocked, but injecting into any other trusted program works.

Reason

Existing Protection Solutions

Lessons

- If you trust a system service, make sure it can’t be tampered with.
- Don’t forget to apply self-protection for features that can be otherwise just disabled.
- Follow and adapt to the OS architecture changes - in our case that’s the introduction of the Frame Server.

Conclusions

A significant privacy threat

Webcam-related Windows internals

Software protection is not perfect

Reason

Sharing is Caring

- A technical paper on the subject:
  Through the looking glass: webcam interception and protection in kernel mode

- A sample driver for blocking access to the camera:
  https://github.com/ReasonSoftware/webcam-interception-driver

- Contact me for any additional information.
  ✉ michael.maltsev@reasonsecurity.com
  🐦 @m417z

Reason

Thank You

Questions?