RED OPS
INFORMATION SECURITY

D3FC0N

How to tamper the EDR?

Master of Puppets

Whoami

Daniel Feichter

- Founder of **RedOps (formerly Infosec Tirol)**
- Originally industrial engineer, since about 4 years passioned, wannabe red teamer
  - Endpoint security on Windows
  - Advanced Persistent Threat emulation
  - Endpoint security research, mostly antivirus & EDR
  - Favourite ATT&CK tactic, Defense Evasion TA0005

- Martial arts fan and fully convinced EDR user
- Twitter @VirtualAllocEx

Disclaimer

- It’s only about my personal experience / journey
- I make no claims to completeness
- No Zero days, just learning about EPP/EDR mechanisms and functionality on Windows
- Shown strategy / concept applies to multiple products on Windows
- Speaking about EDRs, I always refer to EPP/EDR combinations
- Feel free to ask, excluded, which product was used in the demos (vendor neutrality) 😉

We take a look at

- ATT&CK T1562.001: Impair Defenses: Disable or Modify Tools

- Disable main functionalities from EDR, without relying on:
  - EDR uninstall password / token
  - Using any uninstall software
  - Uninstalling EDR in general
  - Using Windows Security Center

- Similar seen in the wild, by AvosLocker Ransomware

We want to achieve

- Deep dive AV/EPP/EDR products on Windows
  - EDR components user space and kernel space
  - Functionality and relationship between user- and kernel space

- Tamper EDR key component, disable EDR and get permanently rid of:

Antivirus capabilities	EDR capabilities	EDR capabilities
Based on user space DLL injection -> user space API hooking	Active response (detections); Telemetry footprint	Host isolation; Real time response; sensor recovery

Give me a scenario

- Red team engagement
  - Initial access: phishing or similar
  - Achieved privileged user rights: exploit or misconfiguration
  - Explore process structure -> additional useful user session open

T1003.001

OS credential dumping: LSASS memory

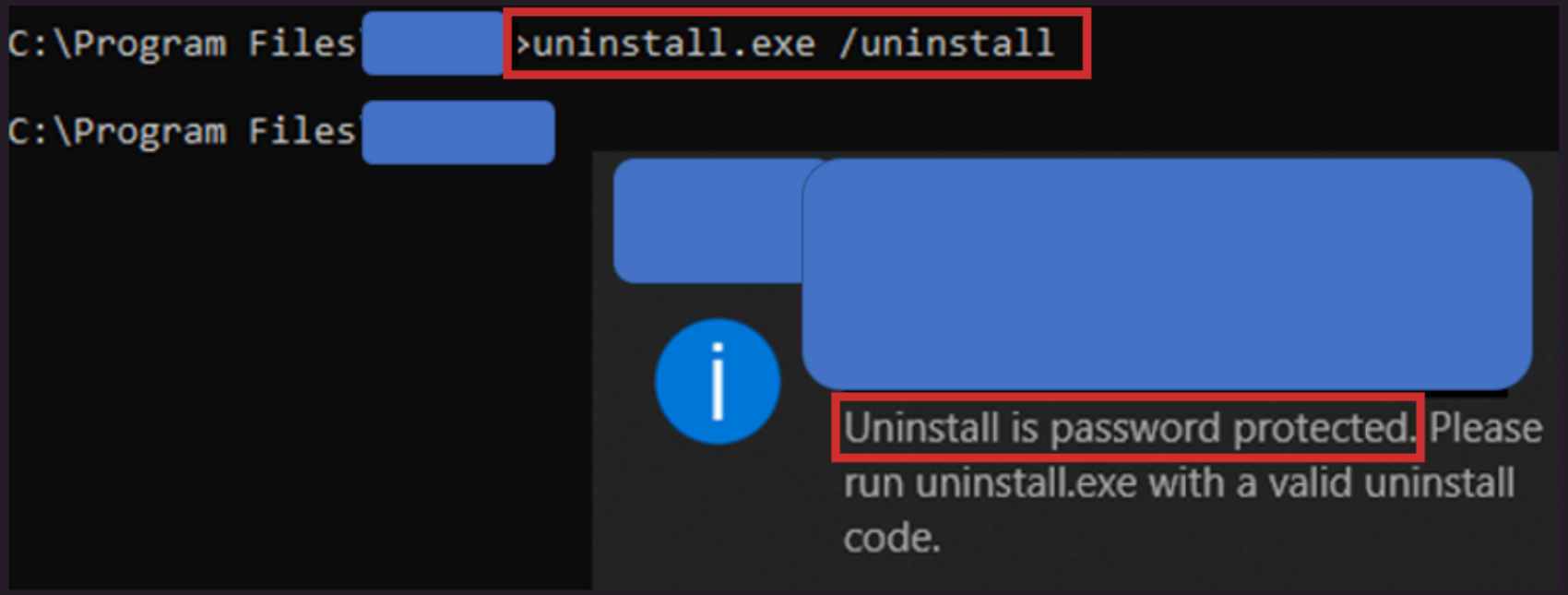
T1134.001

Access token manipulation: token impersonation/theft

- But installed EDR is tough! -> Beginning of my private EDR tampering journey

Come on, I am already admin

- Despite privileged user rights, most EDRs still annoying
- Why not simply uninstall the EDR?



User space

First step: EDR processes

User-space component: EDR processes

- Normally, initialized as Protected Process Light (PPL)
- Despite system integrity, process termination not allowed

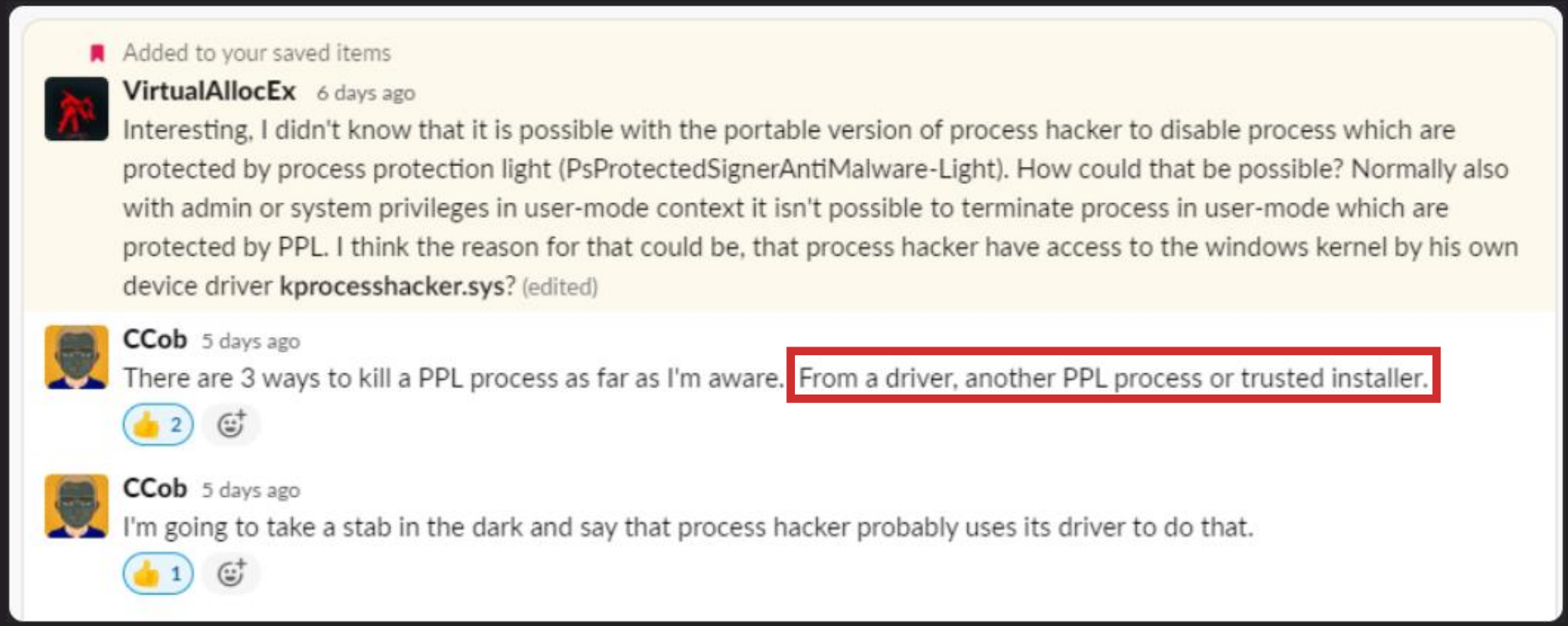
```
C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>taskkill /IM "[redacted]" /F
ERROR: The process "[redacted]" with PID 3296 could not be terminated.
Reason: Access is denied.
```

Process	Protection	User Name	PID
svchost.exe		NT AUTHORITY\NETWORK SERVICE	3260
svchost.exe		NT AUTHORITY\SYSTEM	3288
[redacted]	PsProtectedSignerAntimalware-Light	NT AUTHORITY\SYSTEM	3296
	PsProtectedSignerAntimalware-Light	NT AUTHORITY\SYSTEM	3876
	PsProtectedSignerAntimalware-Light	NT AUTHORITY\SYSTEM	5180
svchost.exe		NT AUTHORITY\LOCAL SERVICE	3340
svchost.exe		NT AUTHORITY\SYSTEM	3440
svchost.exe			

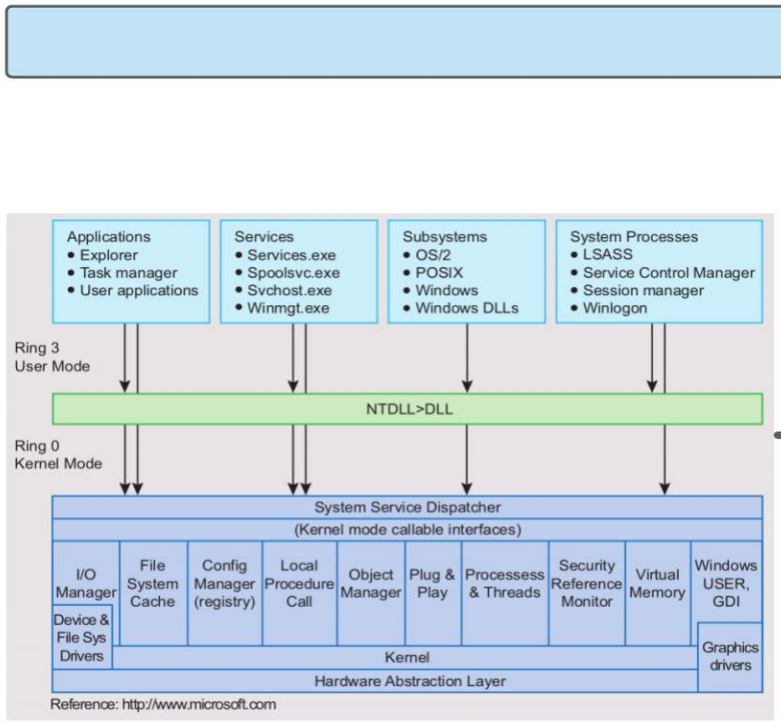
CPU Usage: 3.57% | Commit Charge: 28.43% | Processes: 144 | Physical Usage: 34.83%

EDR processes: disable PPL

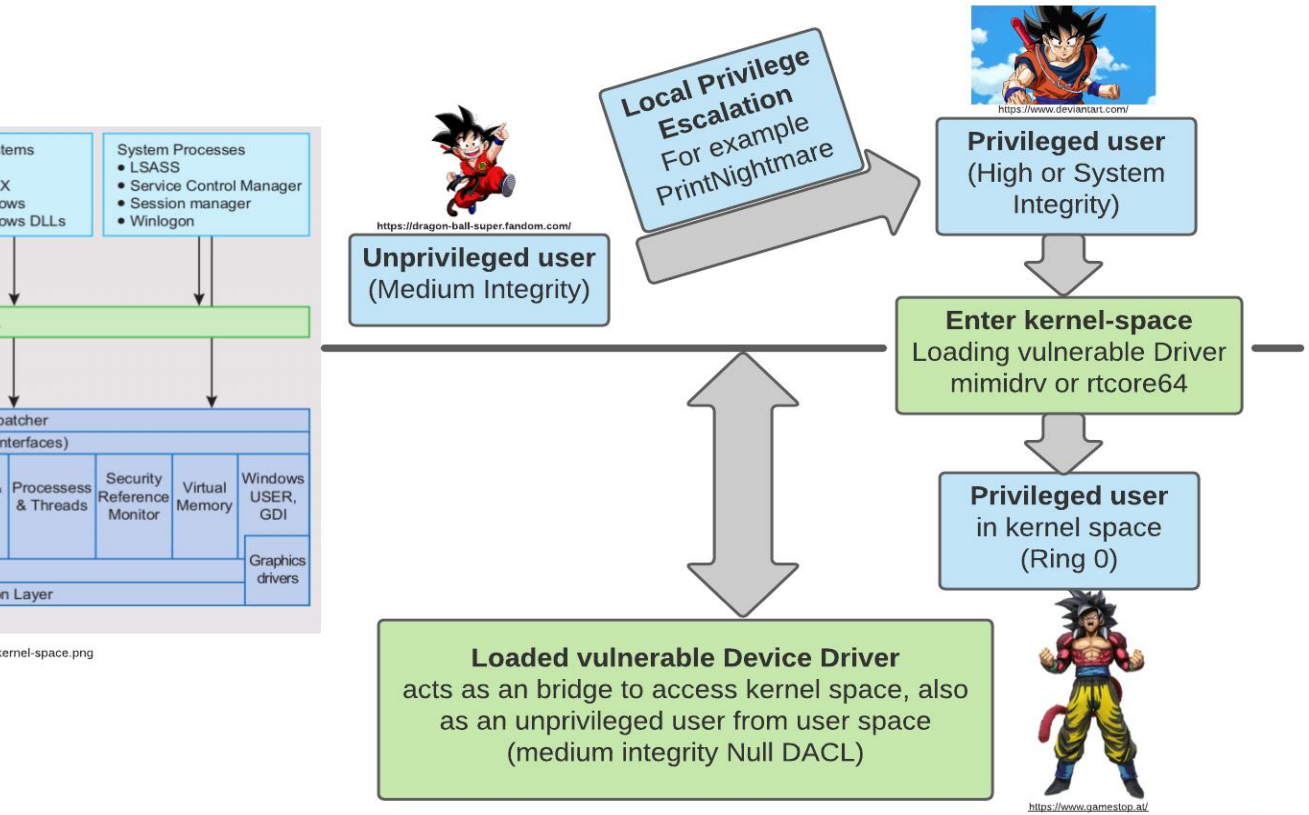
- Signed vulnerable (device) driver -> RTCore64 CVE 2019-16098
- Creds to @EthicalChaos



EDR processes: disable PPL



https://www.aldeid.com/wiki/File:User-kernel-space.png



User-space component: EDR process tampering

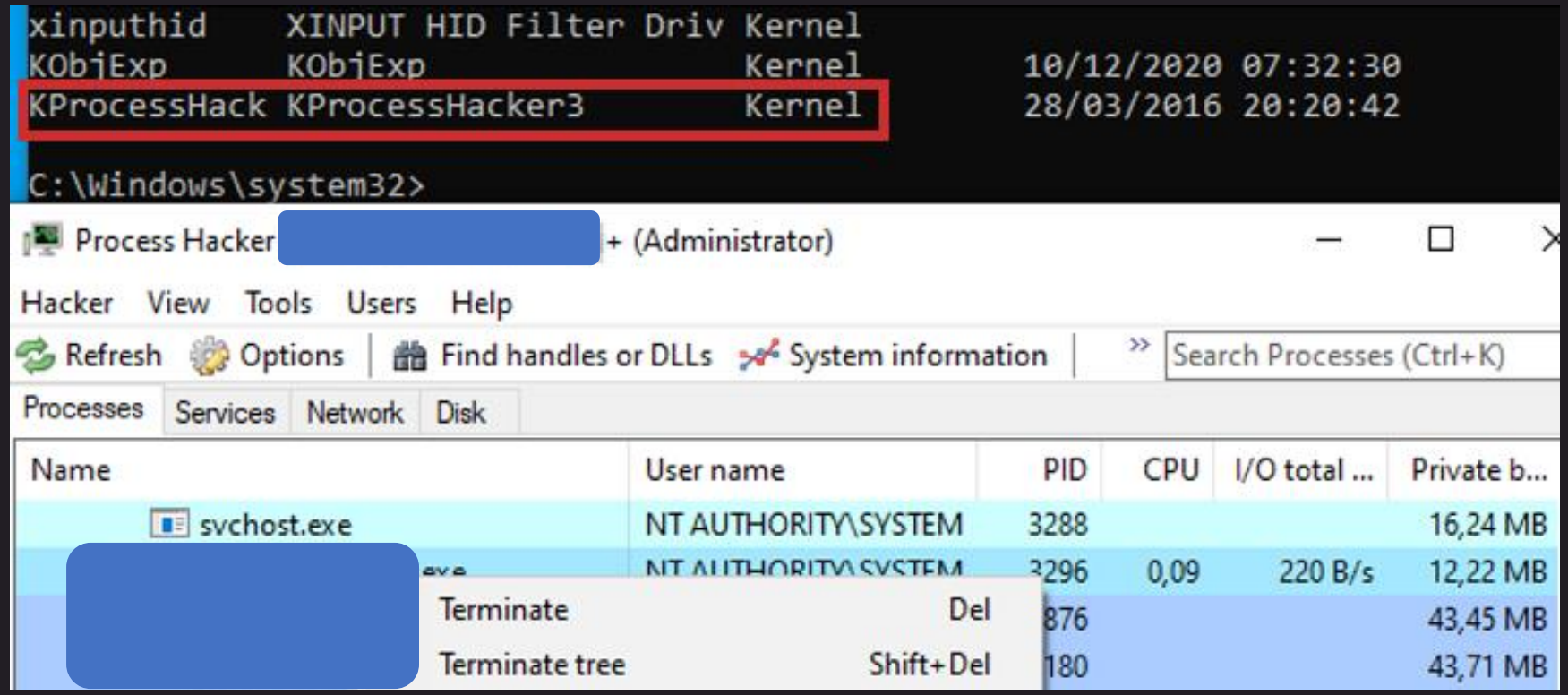
- Tool Time -> PPL Killer -> driver rtcore64.sys or Mimikatz -> mimidrv.sys

```
C:\cache>echo %date% %time%
17/01/2022 15:49:36,76

C:\cache>mimikatz.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 # privilege::debug
Privilege '20' OK

mimikatz # !+
[*] 'mimidrv' service not present
[+] 'mimidrv' service successfully registered
[+] 'mimidrv' service ACL to everyone
[+] 'mimidrv' service started

mimikatz # !processprotect /remove /process:edr_process.exe
```

```
C:\cache>echo %date% %time%
17/01/2022 15:45:12,00

C:\cache>PPLKiller.exe /installDriver
PPLKiller version 0.2 by @aceb0nd
Wrote 14024 bytes to C:\Users\local.admin\AppData\Local\Temp\RTCore64.sys successfully.
[*] 'RTCore64' service not present
[+] 'RTCore64' service successfully registered
[+] 'RTCore64' service ACL to everyone
[+] 'RTCore64' service started

C:\cache>PPLKiller.exe /disablePPL PID agent.exe
```

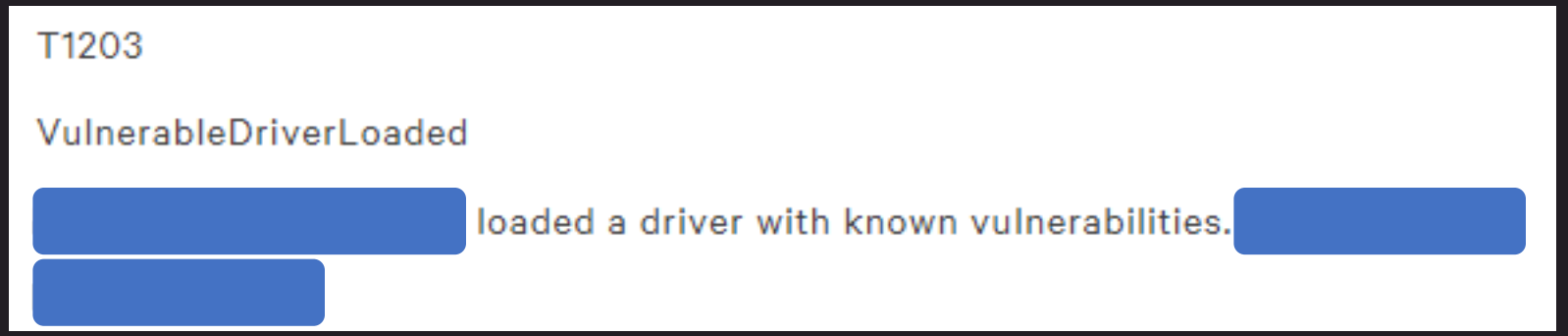
User-space component: EDR process tampering

- Tool Time -> execute Process Hacker as privileged user

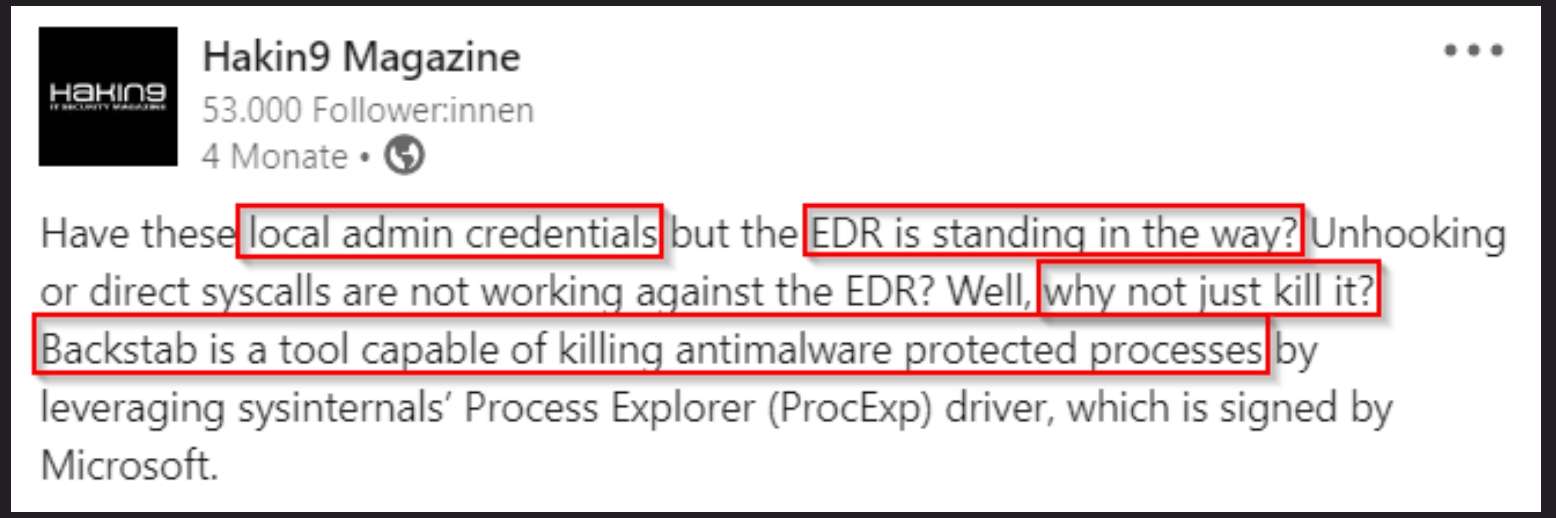


User-space component: EDR process tampering

- EDR vendors start to blacklist / block signed vulnerable drivers
- Depending on product, bypassing is necessary



User-space component: EDR process tampering



Reference: https://www.linkedin.com/feed/update/urn:li:activity:6902622063433986048/

Process termination

Only temporary, gets restarted again and again

Process termination

Even between gap, process terminated and gets restarted EDR works fine

EDR Killed?

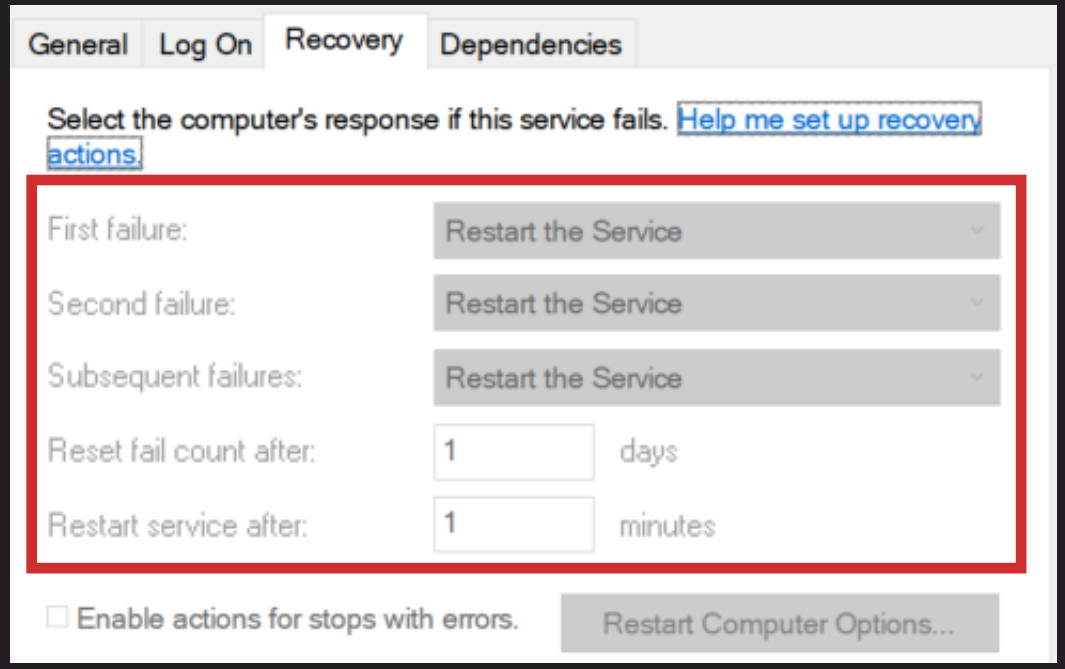
Much to less to get temporary or permanently rid of an EDR!

User space

Second step: EDR services

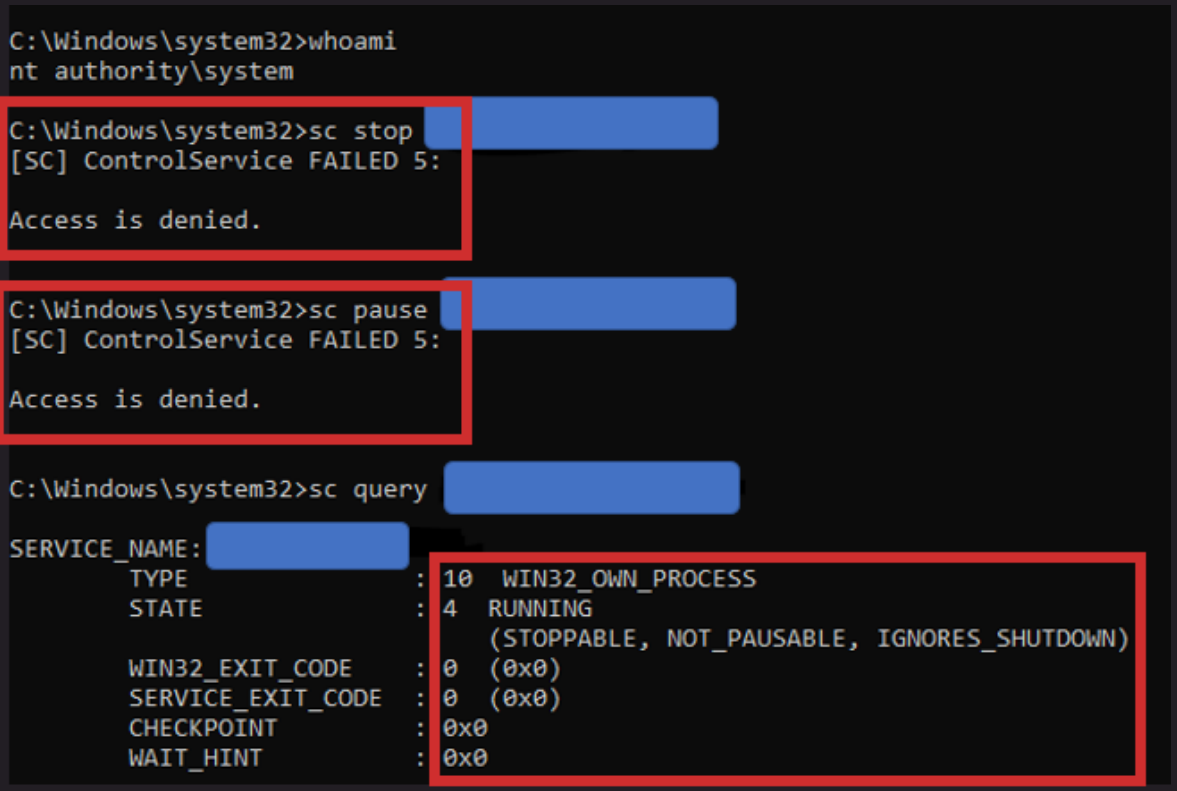
User-space component: EDR service

- Identify EDR service, connected to EDR PPL process
- EDR user space service + EDR user space process = EDR user space component
- Responsible to restart terminated PPL EDR process(es)



User-space component: EDR service tampering

- Initialization as protected service by ELAM driver
- Despite system integrity, not possible (also not temporary) to pause, stop, disable etc.

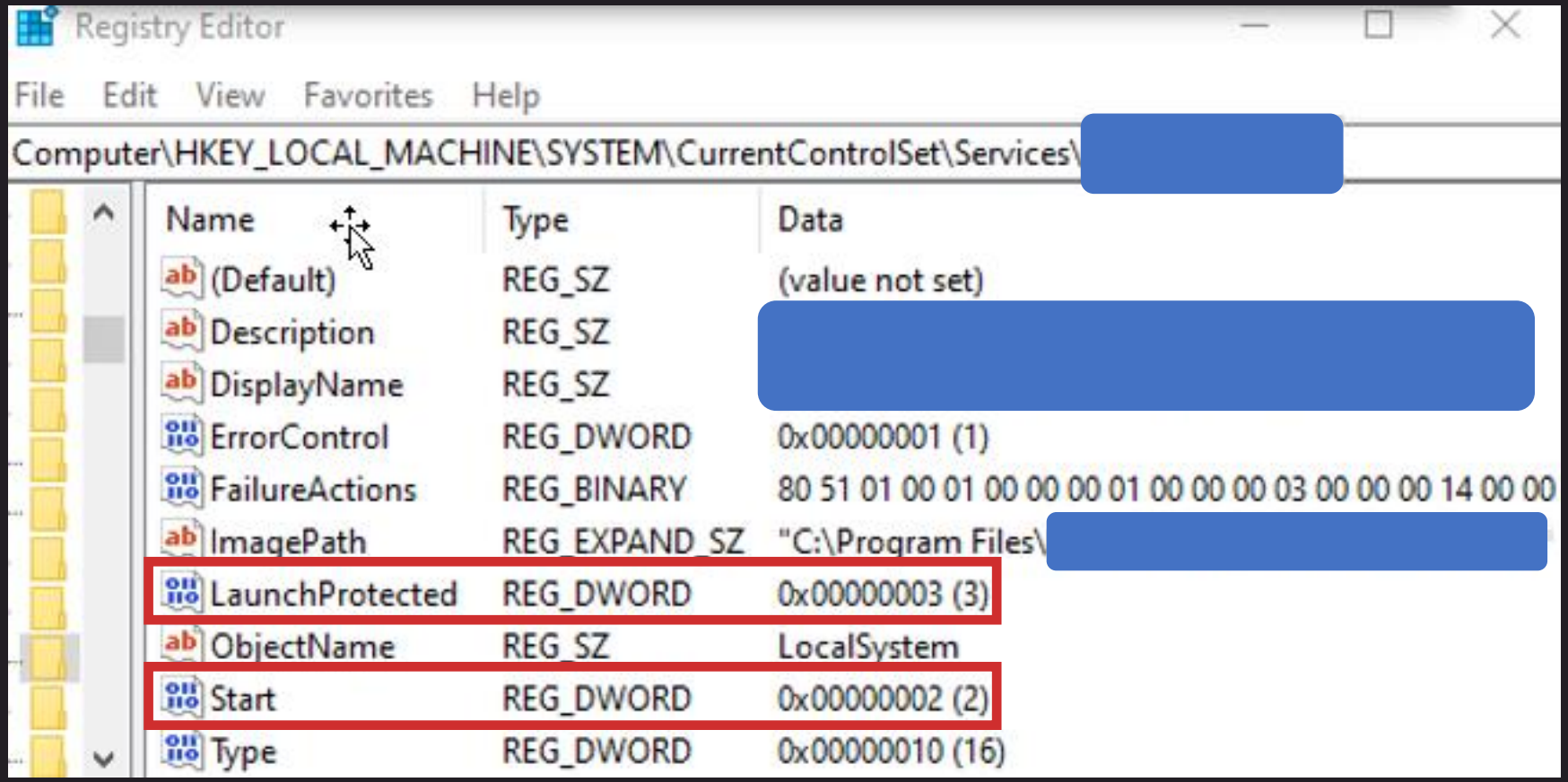


User space

Third step: EDR registry keys

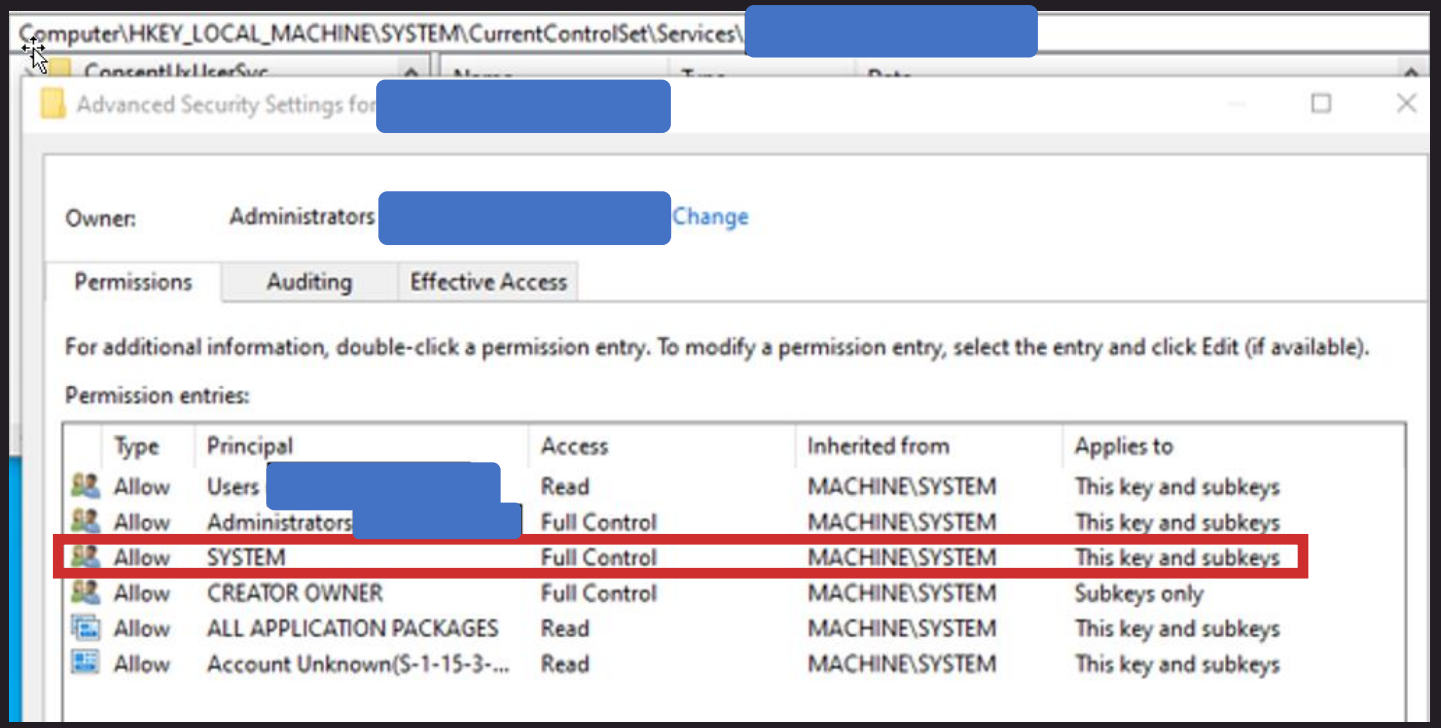
User-space component: EDR registry keys

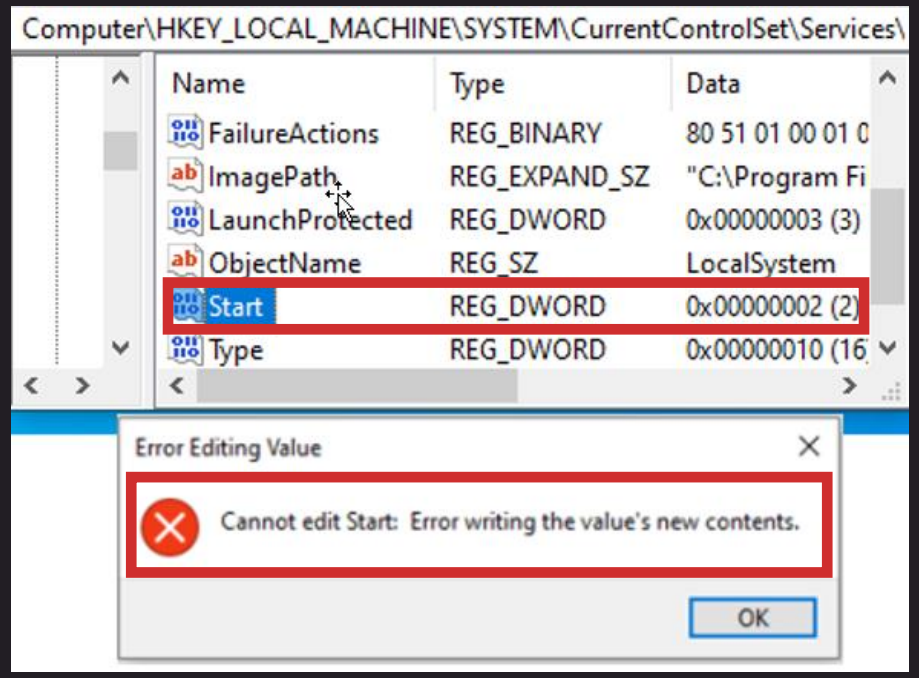
- Identify reg keys / sub keys / entries from EDR user space component (service)



User-space component: EDR registry tampering

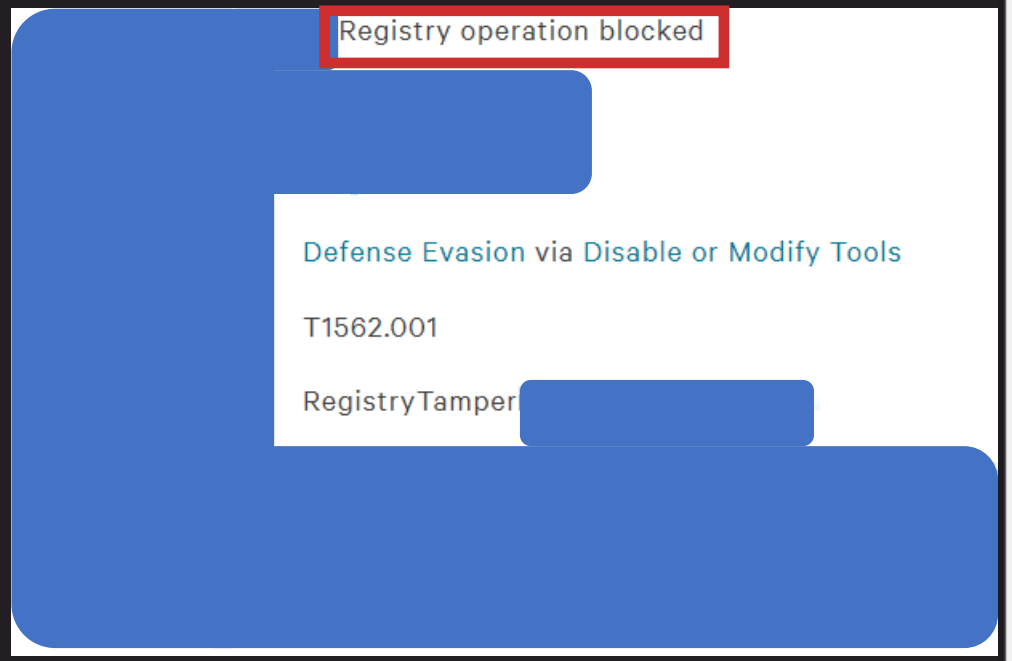
- Start entry: value 2 = autoload and value 4 = disabled
- Tamper reg key -> disable EDR user space component
- Like EDR services and processes, despite system integrity…

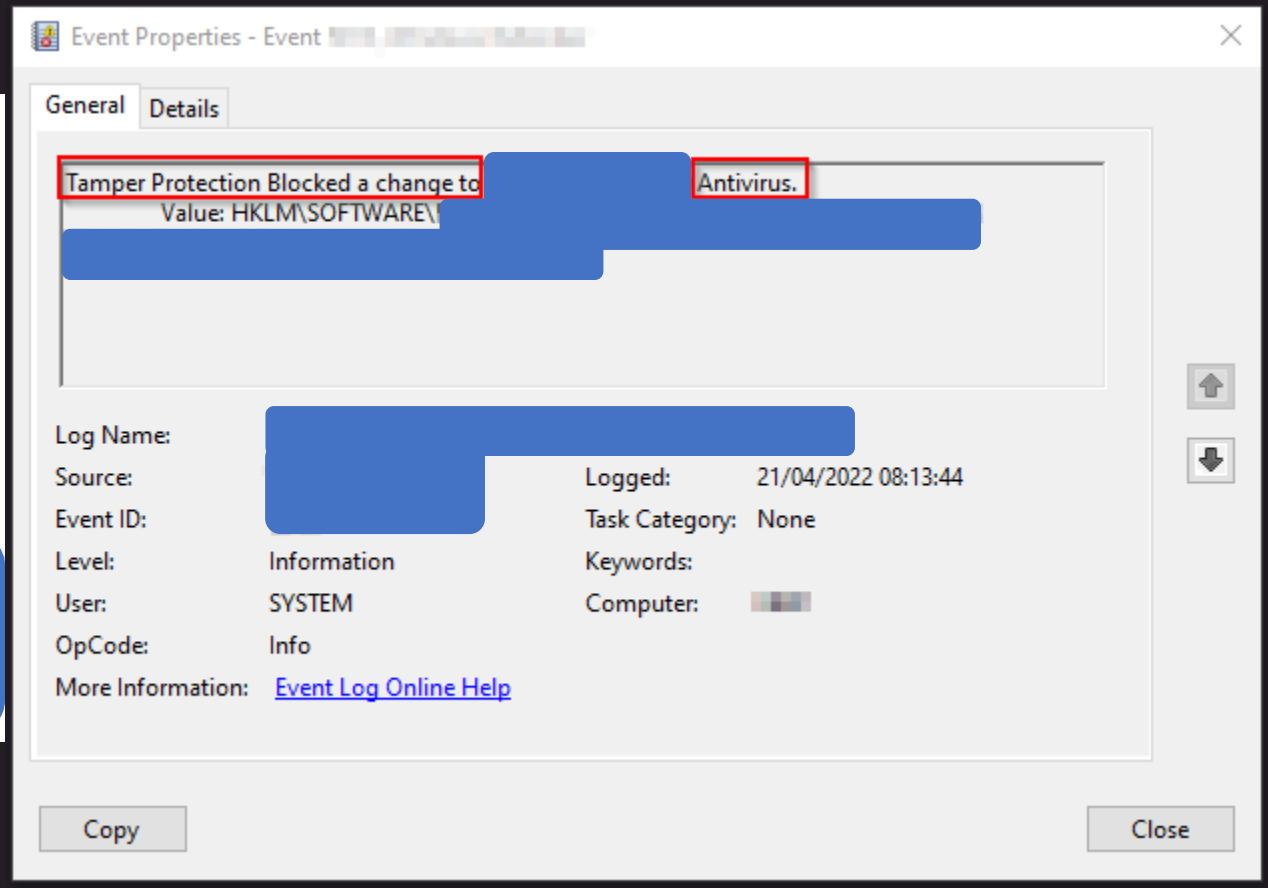




User-space component: EDR registry tampering

- Depending on product -> we (possibly) create tamper protection alerts





Interim status: EDR user space component tampering

EDR processes

Protected by PPL and gets restarted by protected EDR user space service

Current tamper status

- Patch PPL from EDR user space process
- Temporary termination possible

EDR service

Protected by initialization as protected service by EDR ELAM driver

Current tamper status

Compared to EDR processes, also not temporary stoppable or pausable

EDR registry keys

Could be a first key element, but tamper protection until now unknown

Current tamper status

Like EDR services, despite system integrity until now, no tampering possible

Kernel space

Fourth step: EDR kernel callback routines

Kernel-space: EDR callback routines

- Kernel Patch Protection aka PatchGuard
  - (Officially) hooks in kernel space no longer allowed
  - Forced to user space -> user space API hooking
  - Despite Patchguard, different kernel callbacks can be registered:

PsProcessNotifyRoutine	PsThreadNotfifyRoutine	PsLoadImageNotify Routine
User space DLL injection -> user space API-hooking; Telemetry processes	Process Injections	DLL mapping, suspicious image loading

Telemetry collection in general -> attackers footprint based on EDR sensor telemetry

Kernel-space: EDR callback routines

- Besides, used by EDRs to protect their own registry keys against tampering!

On Windows XP, a registry filtering driver can call **CmRegisterCallback** to register a *RegistryCallback* routine and **CmUnRegisterCallback** to unregister the callback routine. The *RegistryCallback* routine receives notifications of each registry operation before the configuration manager processes the operation. A set of **REG_*XXX*_KEY_INFORMATION** data structures contain information about each registry operation. The *RegistryCallback* routine can block a registry operation. The callback routine also receives notifications when the configuration manager has finished creating or opening a registry key.

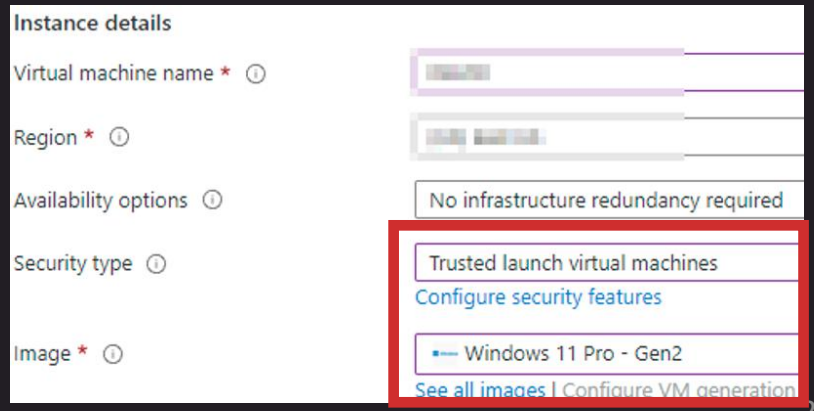
```
                        u_Due_to_Tamper_Protection, blocke 1c000d130     XREF[1]:     FUN_1c0030bf4:1c0030f8d(*)
1c000d130 44 00 75          unicode     u"Due to Tamper Protection, blocked registry d...
          00 65 00
          20 00 74 ...
1c000d1ce 00                ??          00h
1c000d1cf 00                ??          00h

                        u_Due_to_Tamper_Protection, blocke 1c000d1d0     XREF[1]:     FUN_1c003154c:1c00318c9(*)
1c000d1d0 44 00 75          unicode     u"Due to Tamper Protection, blocked registry v...
          00 65 00
          20 00 74 ...
```

First demo: disable EDR user space component

- Using gained knowledge to:
  - Only disable permanently the EDR user space component and what's the impact on:

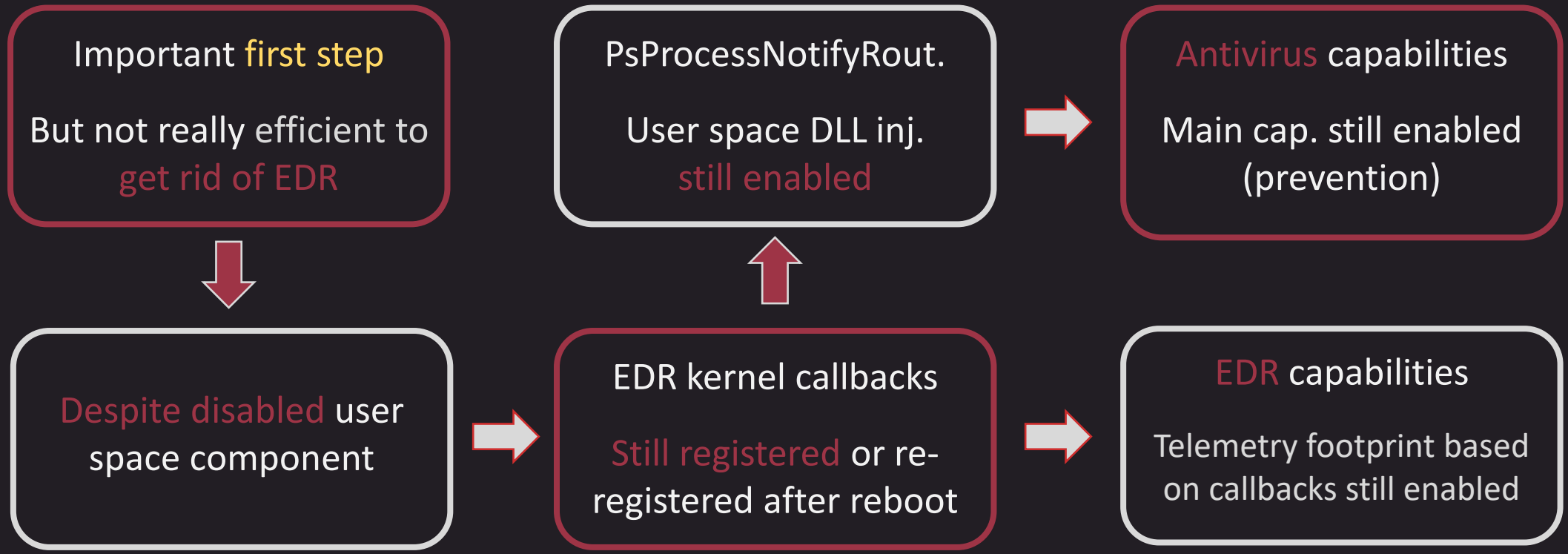
Antivirus capabilities	EDR capabilities	EDR capabilities
Based on user space DLL injection -> user space API hooking	Active response (detections); Telemetry footprint	Host isolation; Real time response; sensor recovery

- **All creds** for the POC CheekyBlinder to @brsn76945860
- Have a look at his amazing blog https://br-sn.github.io/



Conclusion: first demo

- If read/write access kernel space:
  - EDR callbacks can be patched -> registry key tamper protection disabled -> Start entry value 4
  - Disable permanently EDR user space component:



Conclusion: first demo

- If read/write access kernel space:
  - EDR callbacks can be patched -> registry key tamper protection disabled -> Start entry value 4
  - Disable permanently EDR user space component:

Important first step

But not really efficient to get rid of EDR

Despite disabled user space component

EDR capabilities

Still enabled (active), active response, host isolation etc.

Kernel space

Final step: minifilter driver, knockout the EDR!

Kernel-space: EDR minifilter driver

- Independent from EDR user space component
  - Still active, even if EDR user space component is disabled
  - Depending on product, could be responsible for:

Based on the respective callback -> prevention (hooking), detection capabilities (active response and telemetry)

Kernel callback registration in general

EDR web console capabilities

Host isolation, real time response, sensor recovery

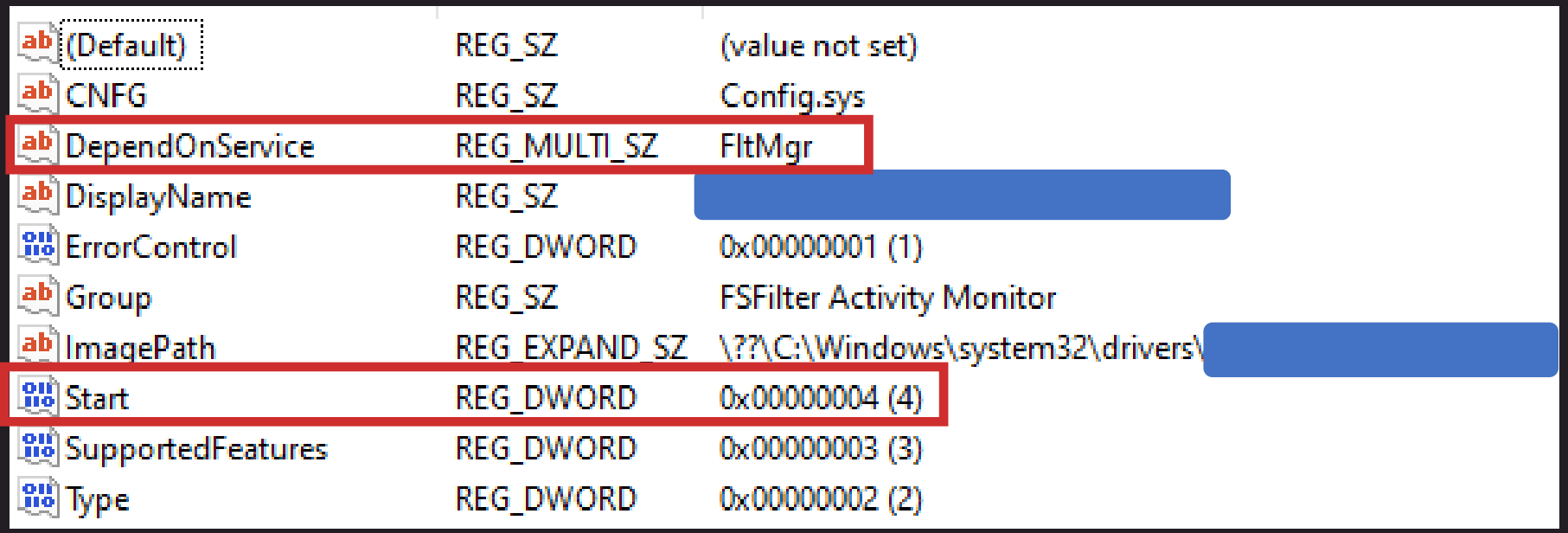
Tampering key element

Permanently get rid of antivirus and EDR capabilities

EDR-minifilter driver (Windows kernel space)

Kernel-space: EDR minifilter driver

- How to disable the EDR minifilter driver?
  - EDR minifilter -> independent registry key
  - Similar structure to EDR user space component reg key -> remember, Start entry value 4



(Default)	REG_SZ	(value not set)
CNFG	REG_SZ	Config.sys
DependOnService	REG_MULTI_SZ	FltMgr
DisplayName	REG_SZ	[redacted]
ErrorControl	REG_DWORD	0x00000001 (1)
Group	REG_SZ	FSFilter Activity Monitor
ImagePath	REG_EXPAND_SZ	\??\C:\Windows\system32\drivers\[redacted]
Start	REG_DWORD	0x00000004 (4)
SupportedFeatures	REG_DWORD	0x00000003 (3)
Type	REG_DWORD	0x00000002 (2)

Second demo: disable EDR minifilter driver

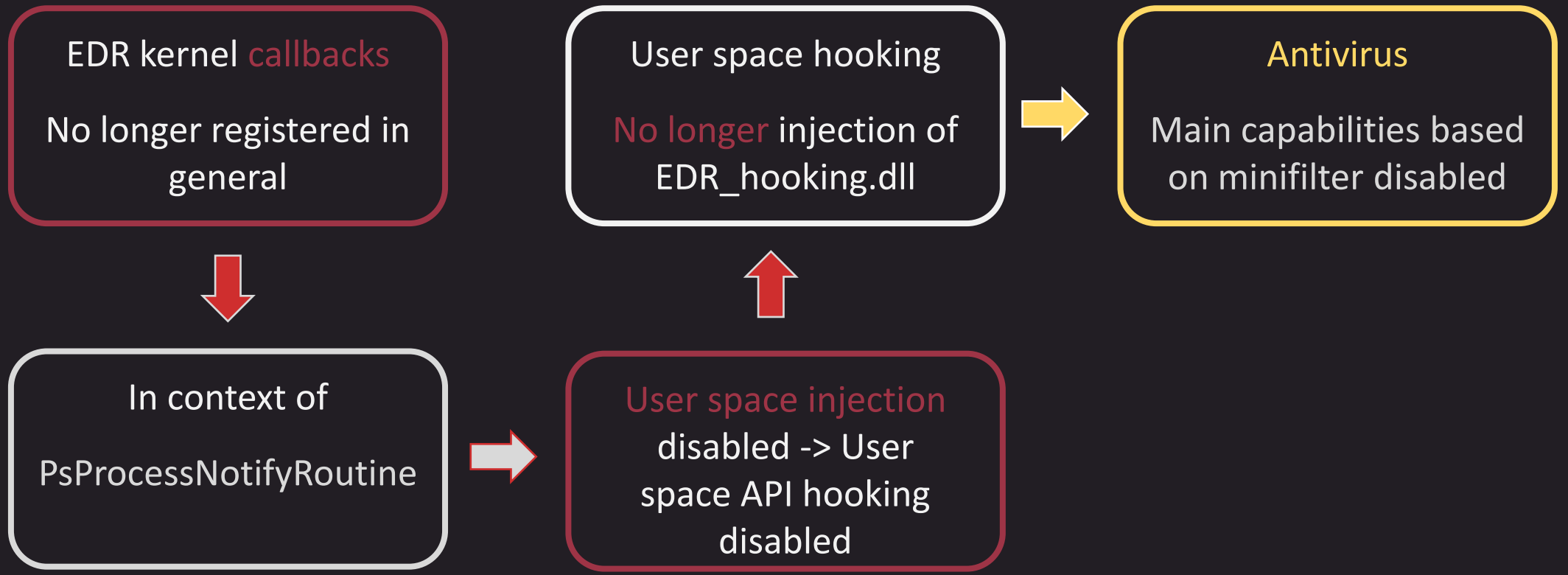
- Using gained knowledge to:
  - Only permanently disable initialization of EDR minifilter driver (kernel component)
  - EDR User space component stays enabled

- What's the impact on:

Antivirus capabilities	EDR capabilities	EDR capabilities
Based on user space DLL injection -> user space API hooking	Active response (detections); Telemetry footprint	Host isolation; Real time response; sensor recovery

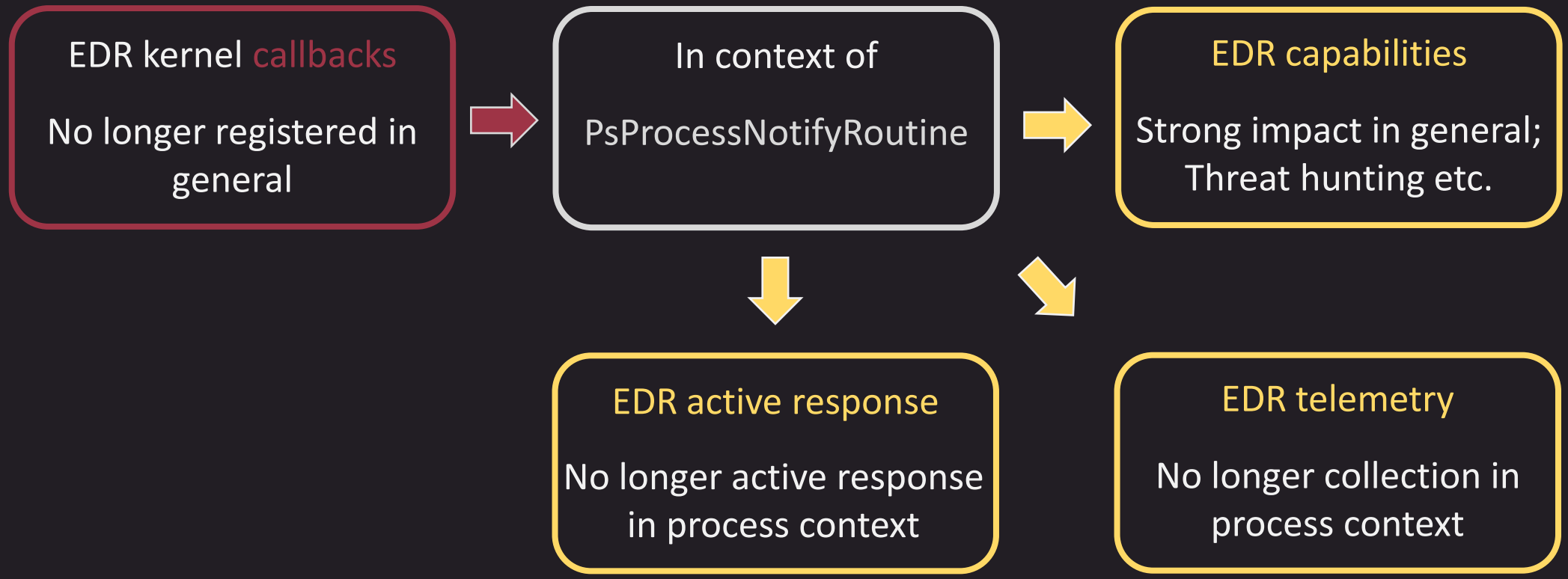
Conclusion: second demo

- Permanently disabling EDR minifilter, much stronger impact:
- Permanently impact on:



Conclusion: second demo

- Permanently disabling EDR minifilter, much stronger impact:
- Permanently impact on:



Conclusion: second demo

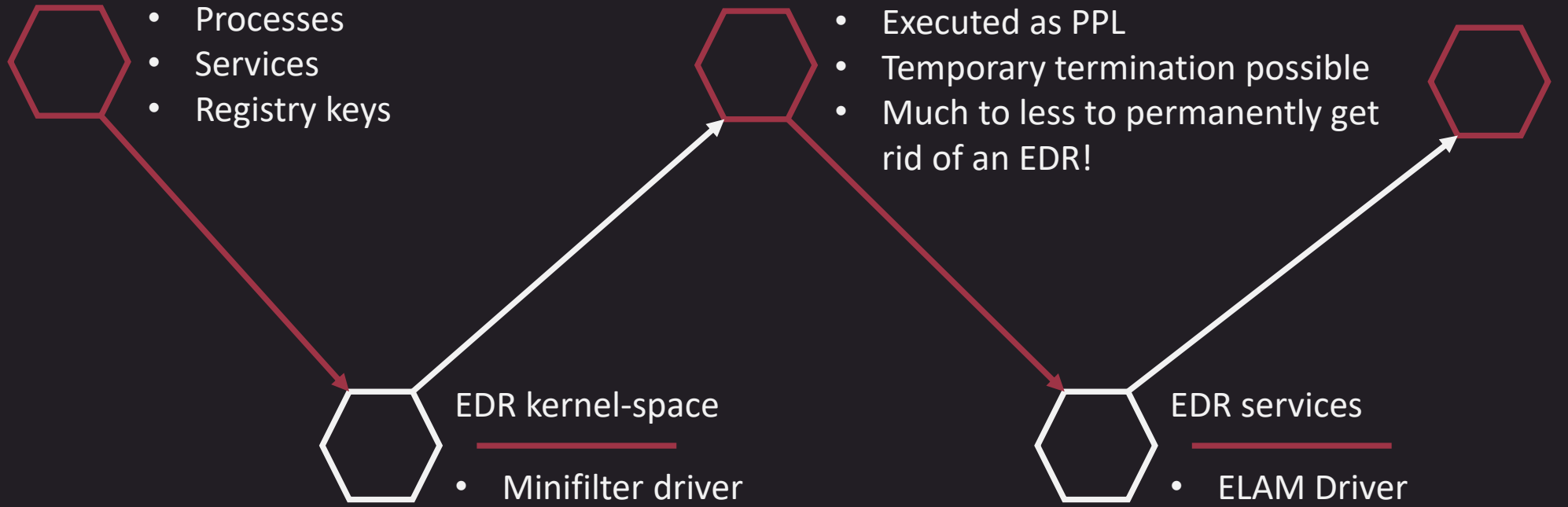
- Permanently disabling EDR minifilter driver, much stronger impact!
  - Disabling the EDR minifilter driver itself:
    - Permanently impact (depending on product) on Blue team EDR web console features

Host isolation	Real time response	EDR sensor recovery
Based on EDR sensor, host isolation no longer possible	Based on EDR sensor, EDR (reverse) shell no longer possible	Based on EDR sensor, recovery of an EDR sensor no longer possible

Summary

End: summary of the talk

Summary



EDR user-space

- Processes
- Services
- Registry keys

EDR kernel-space

- Minifilter driver
- ELAM driver
- Firmware driver etc.

EDR processes

- Executed as PPL
- Temporary termination possible
- Much to less to permanently get rid of an EDR!

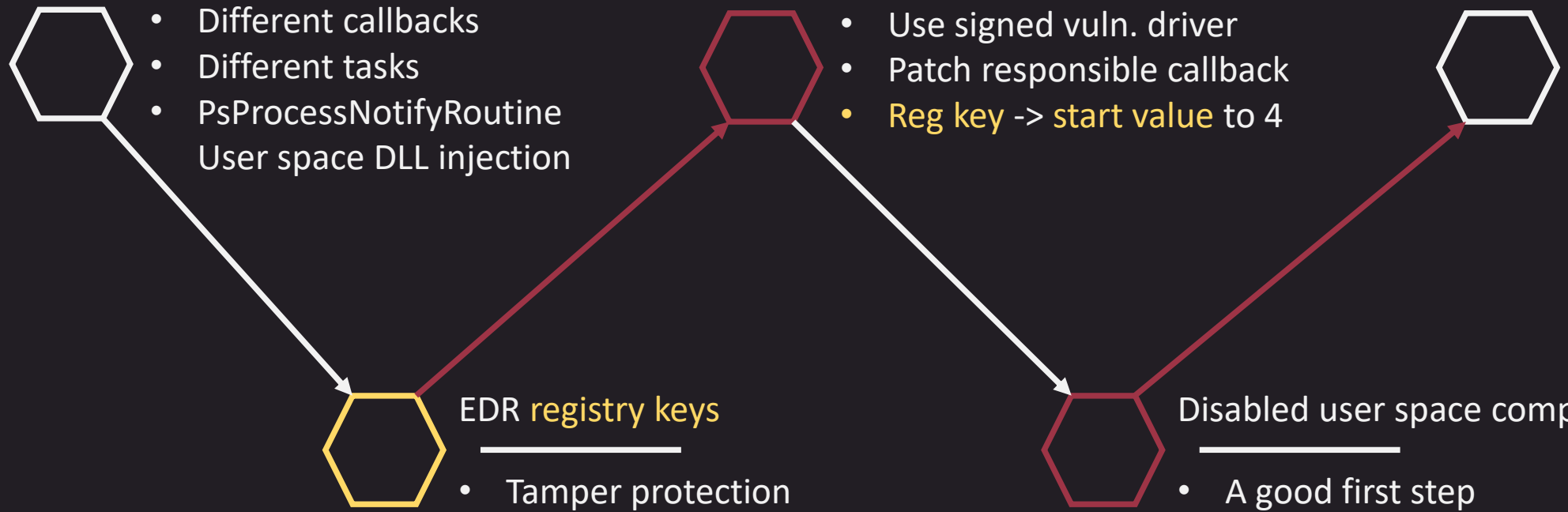
EDR services

- ELAM Driver
- Executed as protected service
- Also, not temporary pauseable stoppable etc.

User-space comp.

- PPL process
- +
- Protected service

Summary



EDR callbacks

- Different callbacks
- Different tasks
- PsProcessNotifyRoutine
  User space DLL injection

Disable user-space comp.

- Use signed vuln. driver
- Patch responsible callback
- Reg key -> start value to 4

EDR minifilter driver

- Independent comp.
- Kernel space
- Responsible for callback registration

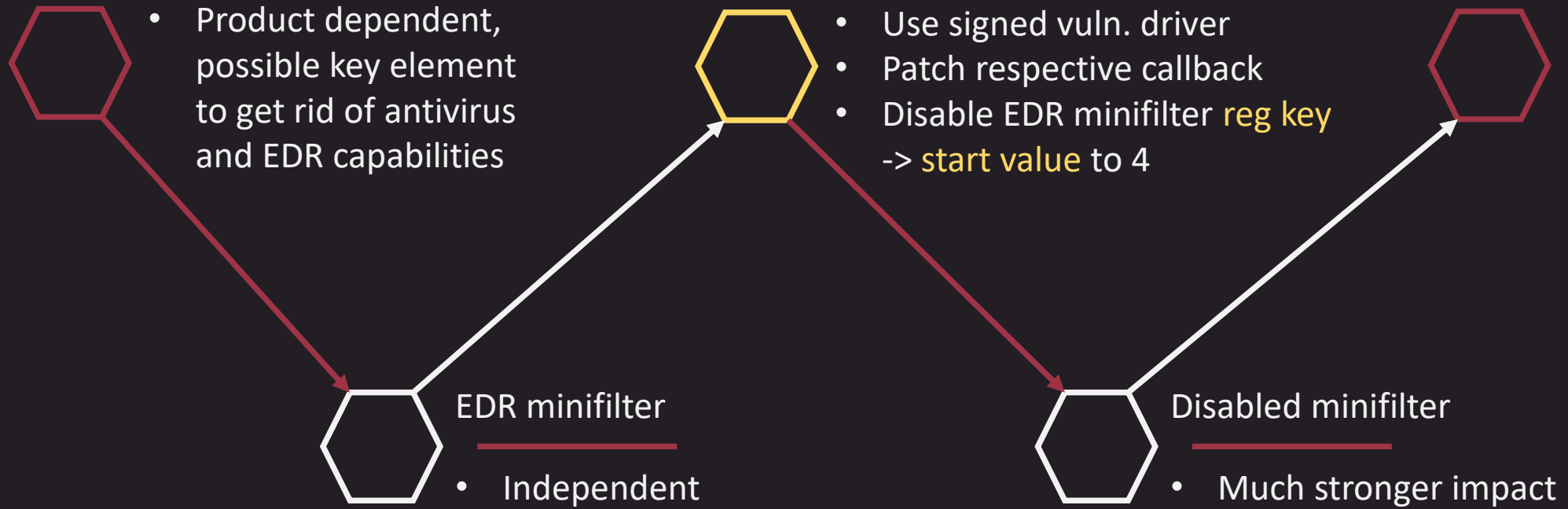
EDR registry keys

- Tamper protection
- Kernel callbacks
- CmRegisterCallback or PsProcessNotifyRoutine

Disabled user space comp.

- A good first step
- But no strong impact on antivirus and EDR capabilities
- Too less to get rid of the EDR

Summary



EDR minifilter

- Product dependent, possible key element to get rid of antivirus and EDR capabilities

EDR minifilter

- Independent protected reg key
- Similar reg key structure compared to user space comp.

Minifilter tampering

- Use signed vuln. driver
- Patch respective callback
- Disable EDR minifilter reg key -> start value to 4

Disabled minifilter

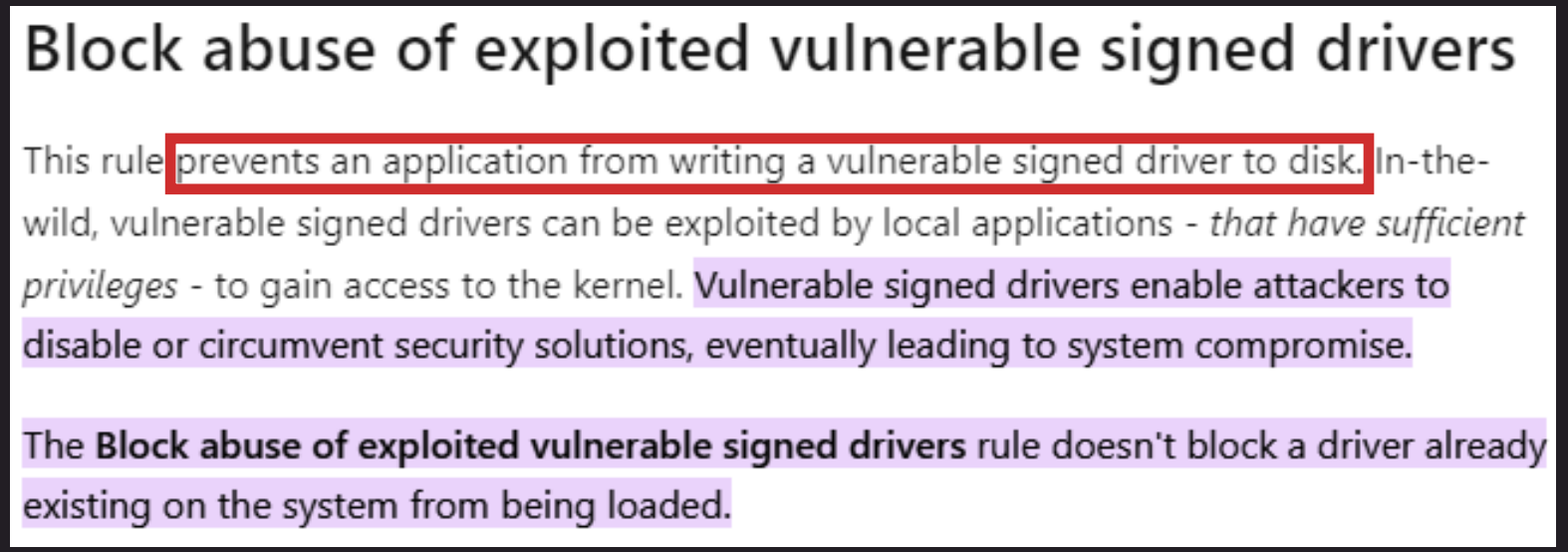
- Much stronger impact compared to disabled user space component
- Permanently get rid of antivirus and EDR capabilities, based on EDR minifilter driver

Conclusion

- Not an EDR vulnerability!
- More a Windows OS Architecture decision
- Same rules for all 3rd party vendors

Blue Team: Mitigation

- Key element is that the attacker get access to kernel space, in case of vulnerable drivers we should try to mitigate this:
- In case of Windows Defender:
  - ASR Rule: Block abuse of exploited vulnerable signed drivers



Block abuse of exploited vulnerable signed drivers

This rule prevents an application from writing a vulnerable signed driver to disk. In-the-wild, vulnerable signed drivers can be exploited by local applications - *that have sufficient privileges* - to gain access to the kernel. Vulnerable signed drivers enable attackers to disable or circumvent security solutions, eventually leading to system compromise.

The **Block abuse of exploited vulnerable signed drivers** rule doesn't block a driver already existing on the system from being loaded.

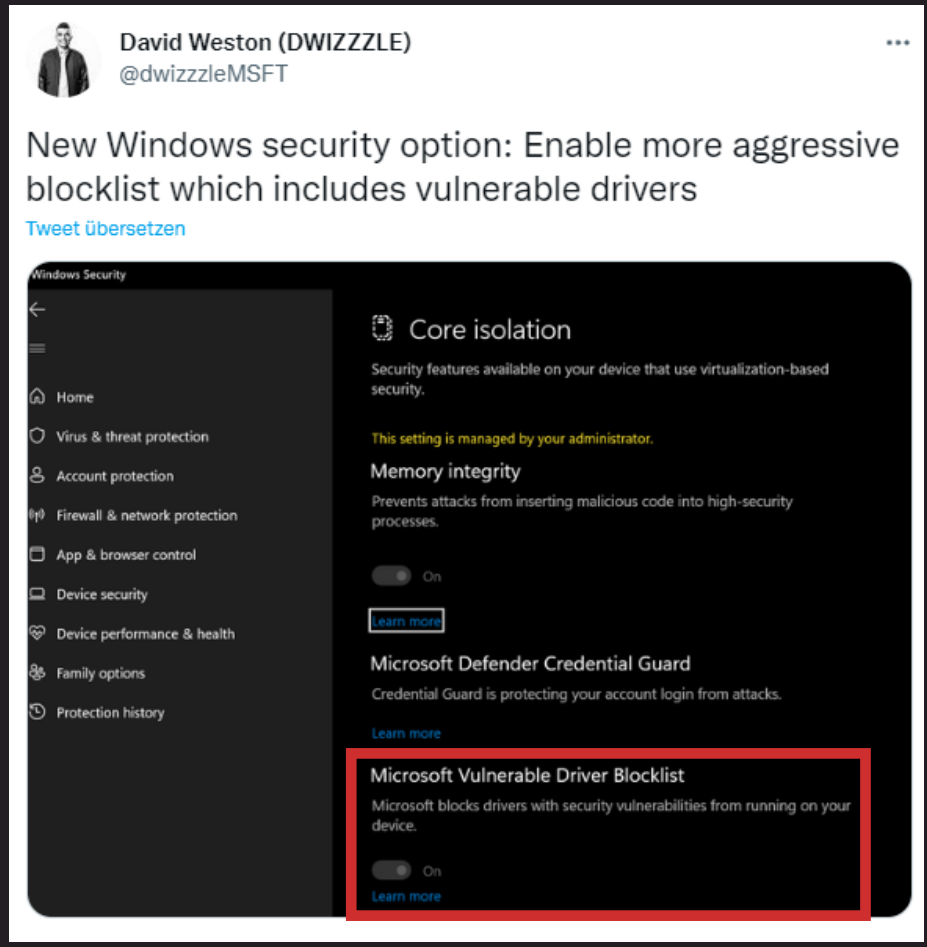
Quelle: **https://docs.microsoft.com/en-us/microsoft-365/security/defender-endpoint/attack-surface-reduction-rules-reference?view=o365-worldwide**

Blue Team: Mitigation

- Windows Device Guard VBS/HVCI:
  - Microsoft Vulnerable Driver Blocklist
  - More aggressive additional hardening with WDAC

> Organizations that want a more aggressive block list than Microsoft's measured approach can add their own drivers to the list using the WDAC Policy Wizard.

Resource: https://www.techrepublic.com/article/how-microsoft-blocks-vulnerable-malicious-drivers-defender-third-party-security-tools-windows-11/



Resource: https://twitter.com/dwizzzleMSFT/status/1508217367259611142

Thank you Las Vegas!

- **Thanks** for the amazing opportunity to be a part of **Defcon 30 / Adversary Village** and thanks to the **greatest community!**
- Thanks to my girlfriend **Brigitte** and my sister **Stefanie** for the **unique support!**
- Check out the blog post https://www.infosec.tirol/how-to-tamper-the-edr/

References

[1] Yosifovich, Pavel; Ionescu, Alex; Solomon, David A.; Russinovich, Mark E. (2017): Windows internals. Part 1: System architecture, processes, threads, memory management, and more. Seventh edition. Redmond, Washington: Microsoft Press. http://proquest.tech.safaribooksonline.de/9780133986471.

[2] Pavel Yosifovich (2019): Windows 10 System Programming, Part 1: CreateSpace Independent Publishing Platform.

[3] Microsoft (2017): Filtering Registry Calls. https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/filtering-registry-calls.

[4] Microsoft (2018): CmRegisterCallbackEx function (wdm.h). https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nc-wdm-ex_callback_function

[5] Microsoft (2018): CmUnRegisterCallback function (wdm.h). https://docs.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/nf-wdm-cmunregistercallback.

[6] @Truneski (2020): Windows Kernel Programming Book Review. https://truneski.github.io/blog/2020/04/03/windows-kernel-programming-book-review/

[7] Matteo Malvica (2020): Silencing the EDR. How to disable process, threads and image-loading detection callbacks https://www.matteomalvica.com/blog/2020/07/15/silencing-the-edr/.

[8] Matteo Malvica (2020): Kernel exploitation: weaponizing CVE-2020-17382 MSI Ambient Link driver https://www.matteomalvica.com/blog/2020/09/24/weaponizing-cve-2020-17382/

[9] Christopher Vella (2020): EDR Observations. https://christopher-vella.com/2020/08/21/EDR-Observations.html.

[10] BR-SN (2020): Removing Kernel Callbacks Using Signed Drivers. https://br-sn.github.io/Removing-Kernel-Callbacks-Using-Signed-Drivers/

References

[11] https://github.com/SadProcessor/SomeStuff/blob/master/Invoke-EDRCheck.ps1

[12] https://synzack.github.io/Blinding-EDR-On-Windows/

[13] https://github.com/SadProcessor/SomeStuff/blob/master/Invoke-EDRCheck.ps1

[14] https://docs.microsoft.com/en-us/windows/win32/api/winsvc/ns-winsvc-service_launch_protected_info

[15] https://sourcedaddy.com/windows-7/values-for-the-start-registry-entry.html

[16] https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/types-of-windows-drivers

[17] https://courses.zeropointsecurity.co.uk/courses/offensive-driver-development

[18] https://www.ghacks.net/2022/03/28/windows-defender-vulnerable-driver-blocklist-protects-against-malicious-or-exploitable-drivers/

[19] https://www.techrepublic.com/article/how-microsoft-blocks-vulnerable-malicious-drivers-defender-third-party-security-tools-windows-11/

[20] https://github.com/eclypsium/Screwed-Drivers/blob/master/presentation-Get-off-the-kernel-if-you-cant-drive-DEFCON27.pdf

[21] https://www.bsi.bund.de/DE/Service-Navi/Publikationen/Studien/SiSyPHuS_Win10/AP7/SiSyPHuS_AP7_node.html