



Memory Forensics and DMA Attacks with MemProcFS and PCILeech

2019-10-23

 UlfFrisk

MemProcFS: Fast easy-to-use Memory Analysis
PCILeech: Direct Memory Access (DMA) Attack Software

Agenda

DMA-attacks with **PCILeech**
PWN LINUX and **WINDOWS** with DMA code injection
MOUNT live file systems
Spawn System **Shell**

Memory Forensics with **MemProcFS**

What is **The Memory Process File System?**
In-Depth: Capabilities Design, API and Plugins

Demos - **Live Demos!**

whoami: Ulf Frisk

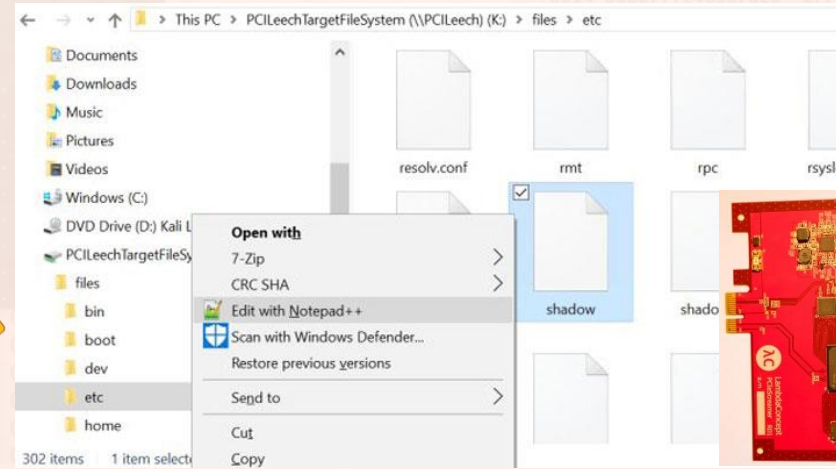
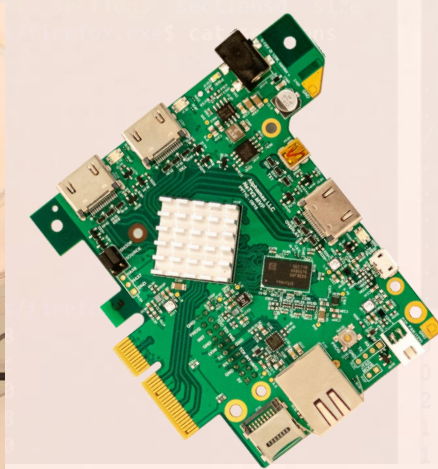
Pentester by day @ Polisen IT – Stockholm, Sweden

Security Researcher by night

Author of the PCILeech Direct Memory Access Attack Toolkit

Presented at different cons including DEF CON, BlueHat and the CCC

100% Open Source

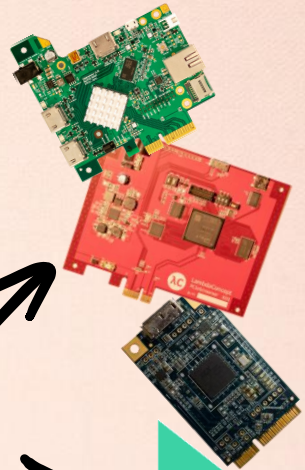


PCILeech and MemProcFS

PCILeech

MemProcFS

LeechCore Library



Dumplt.exe

RAW

DMP

HV
SAVE

CORE



PCILeech

A New Hardware: The NeTV2

github.com/ufrisk/PCILeech

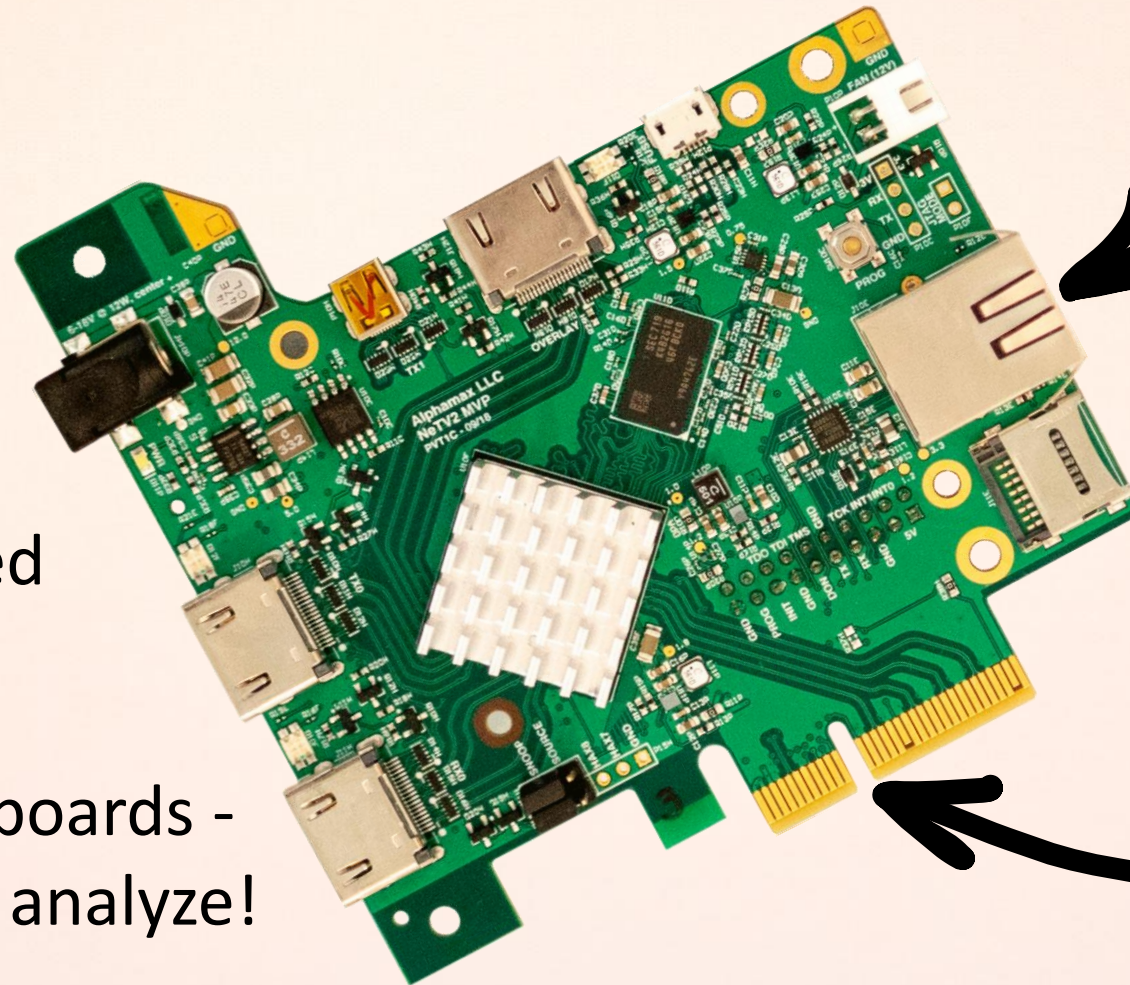
The NeTV2

100Mbit ETH
~8MB/s DMA

\$215

Cheapest Supported
FPGA Hardware

slower than other boards -
plenty for attack & analyze!



PCIe



PCILeech

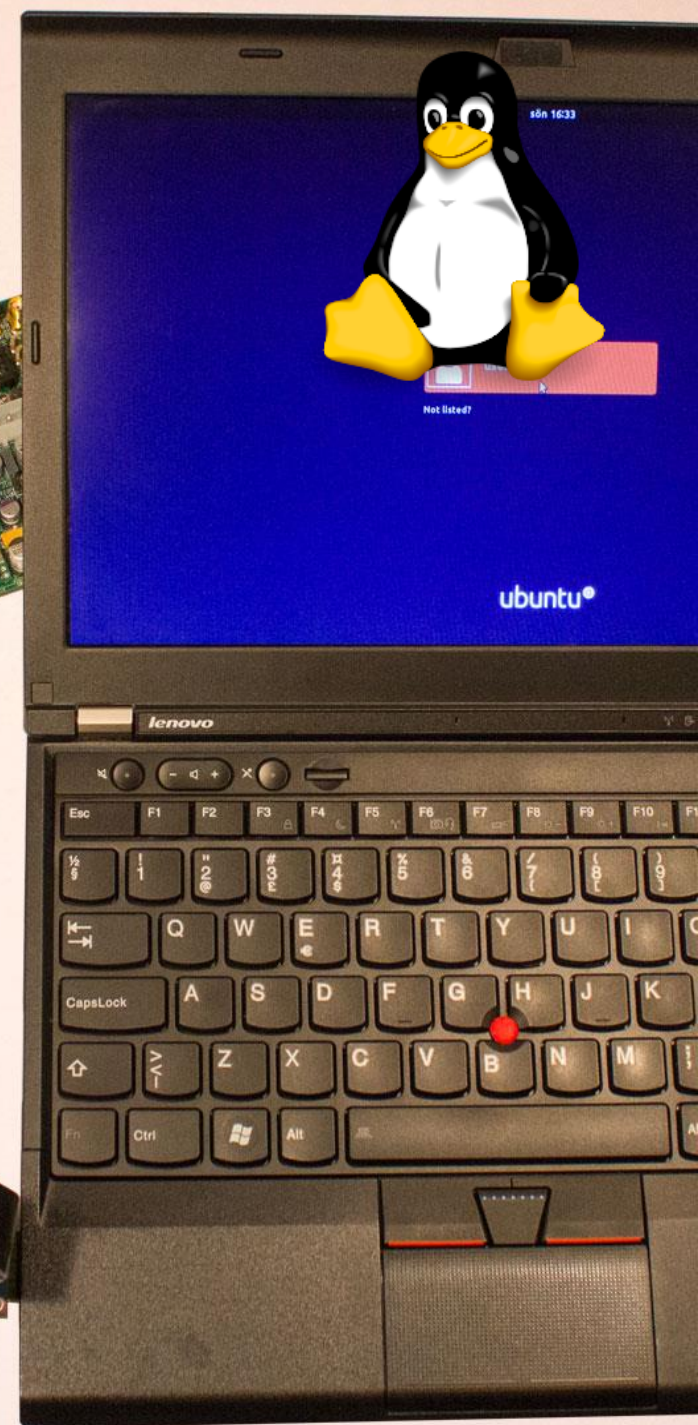
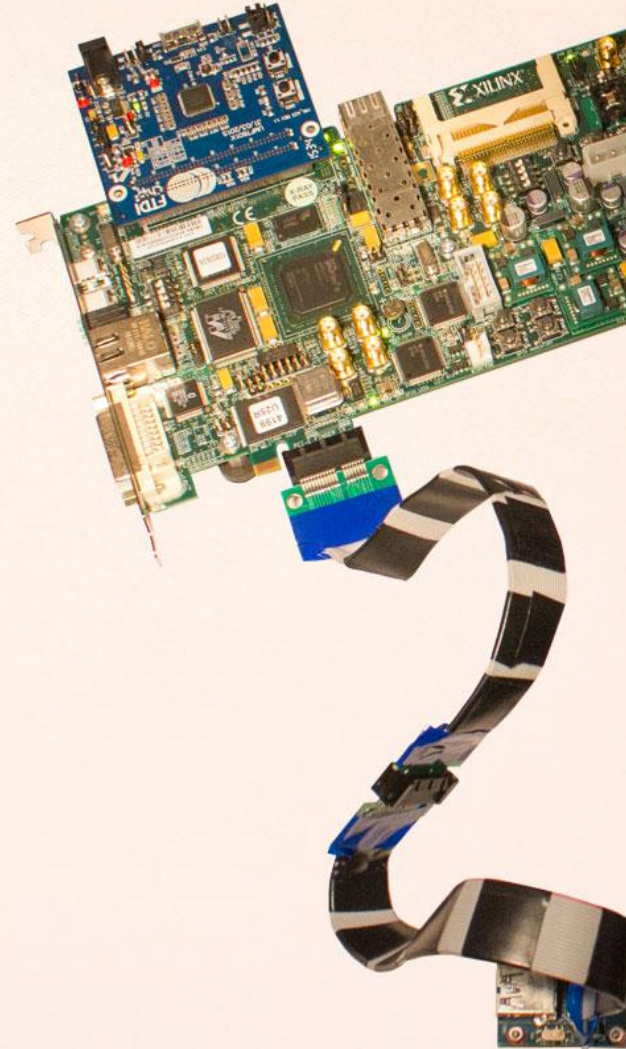
DEMO: LINUX

github.com/ufrisk/PCILeech

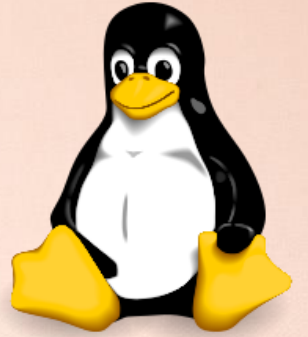
Linux DEMO

KERNEL IMPLANT MOUNT file system UNLOCK

Target System: Kali 2019.3



Linux Security



Hardware without DMA ports

BIOS password, DMA port lockdown, Pre-boot authentication

IOMMU / VT-d (virtualization-based security)



== secure by default!

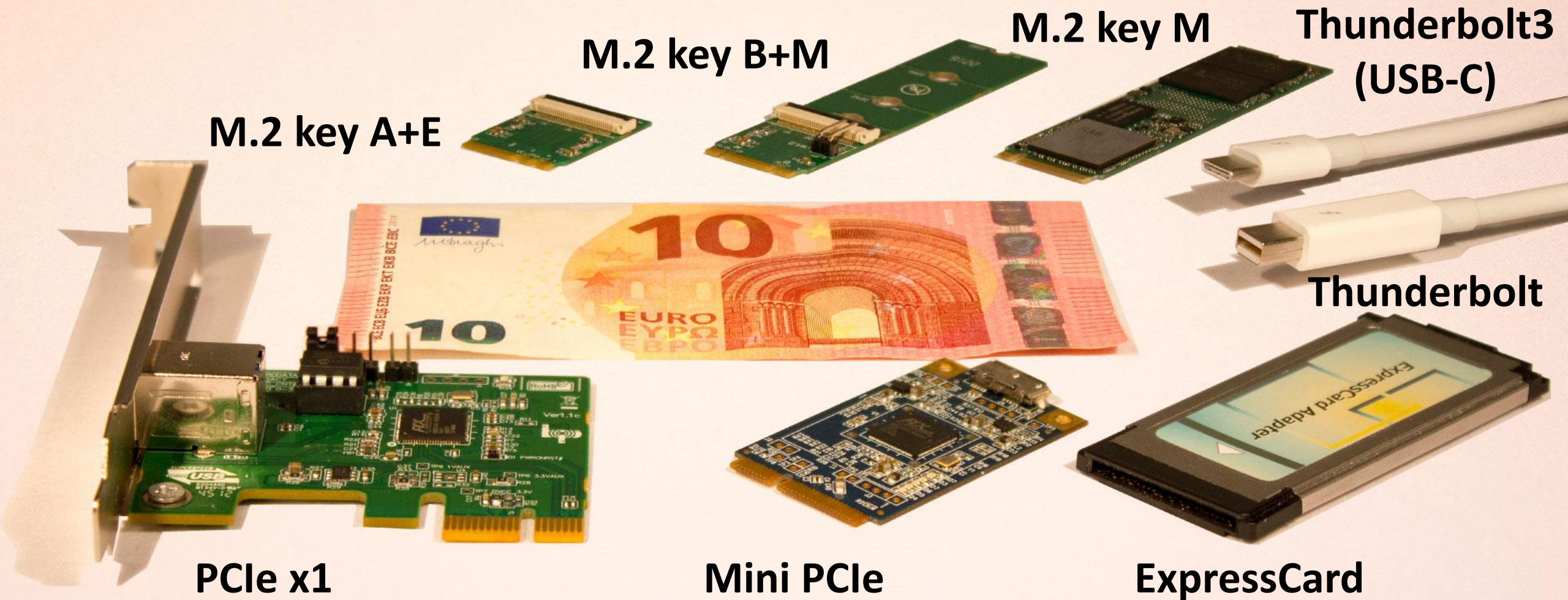


PCILeech

PCI EXPRESS DIRECT MEMORY ACCESS – DMA

github.com/ufrisk/PCILeech

PCI Express Form Factors

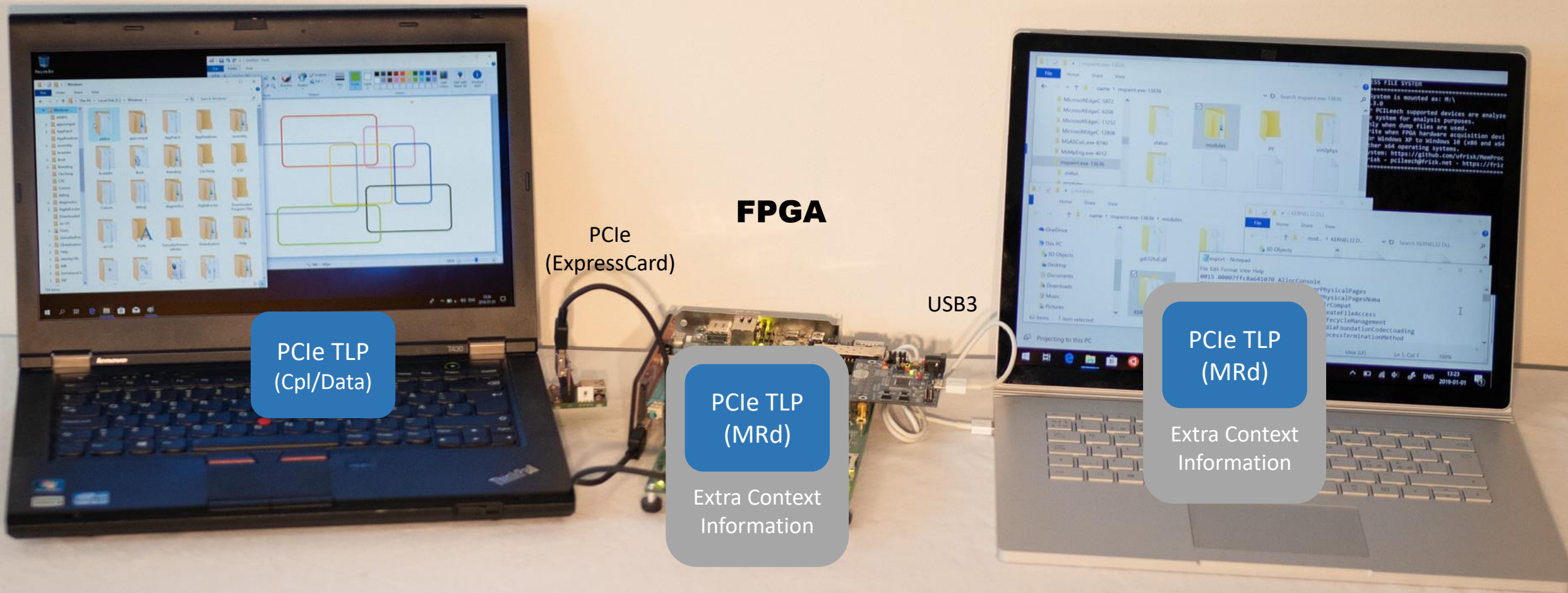


Everything here is PCI Express in different form factors and variations.

Attack & Analysis with DMA HW device

Target Computer

Analysis Computer





PCILeech

DEMO: WINDOWS

github.com/ufrisk/PCILeech

Windows DEMO:

Execute Code and
Spawn Shell

MOUNT:
target file system

Target System: Windows 10 1903



Windows Security:

Win7 = vulnerable always!

Win10 = (in)secure depending on hardware & config

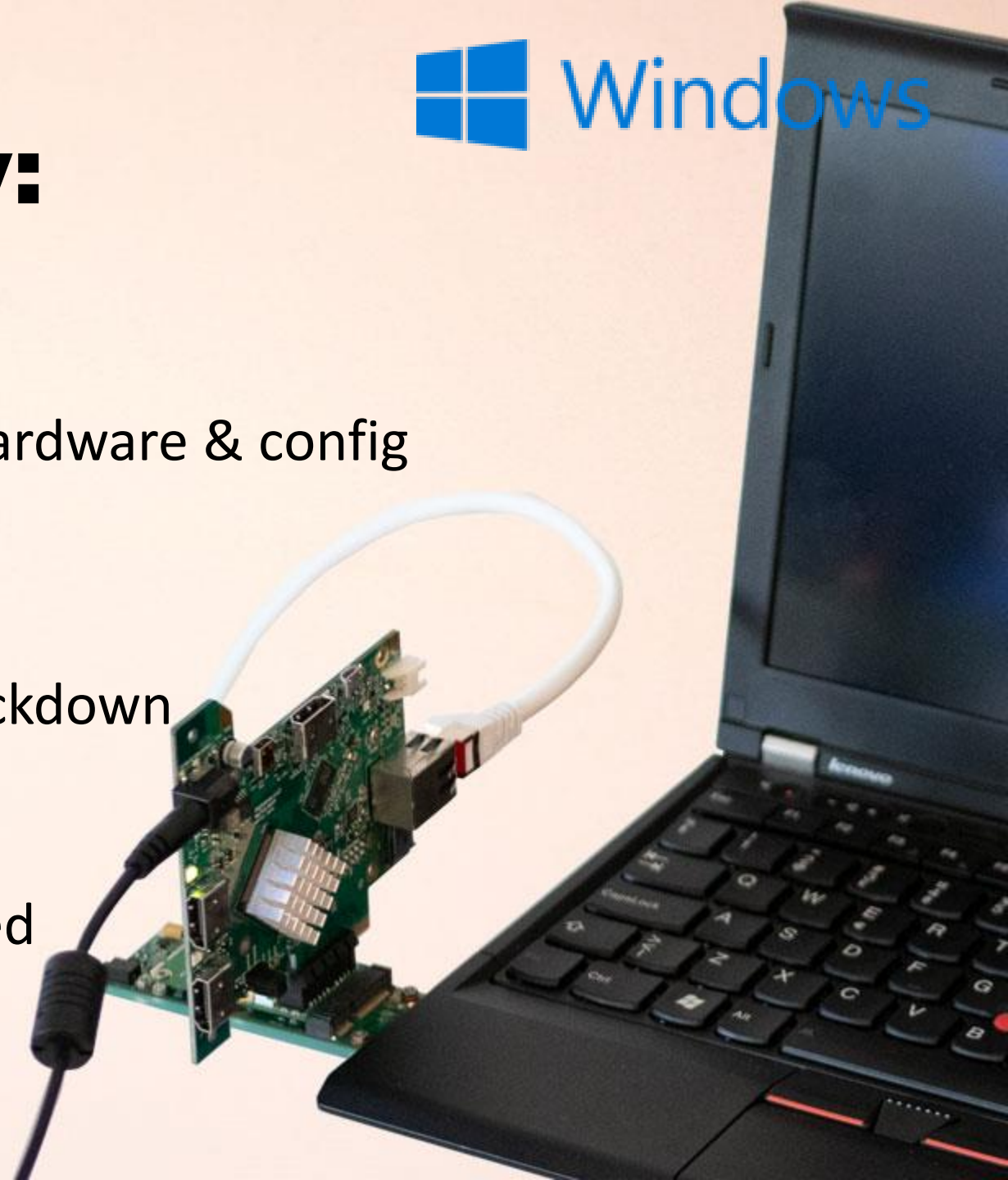
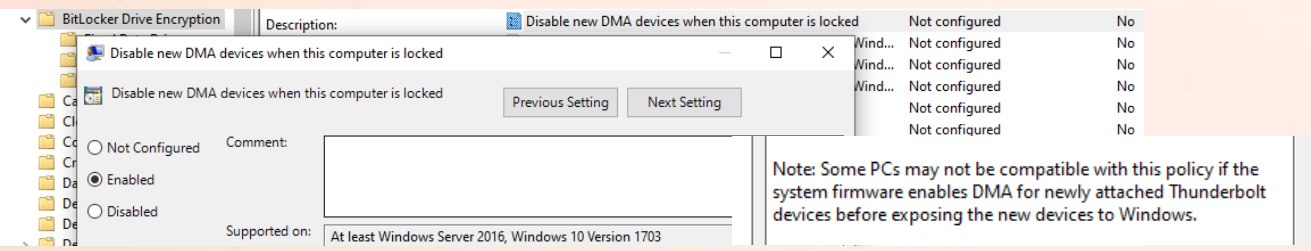
Hardware without DMA ports

Bitlocker PIN, BIOS password, port lockdown

IOMMU / VT-d,

Virtualization based security,

Disable new DMA devices when locked



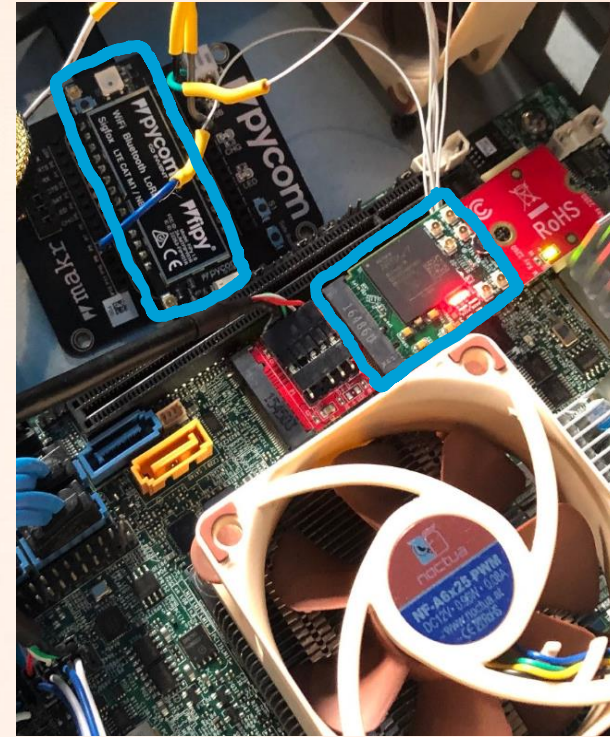
PCILeech: Future

More supported devices

Access via WiFi/4G

Windows User-Mode Implants

Research: macOS/Windows/Linux/ChromeOS/Android/ARM



BLACKHAT USA 2019

**PICODMA:
DMA ATTACKS
AT YOUR
FINGERTIPS**



MemProcFS

The Memory Process File System

github.com/ufrisk/MemProcFS

MemProcFS - The Memory Process File System

Memory Analysis tool with **Windows** focus

In-Memory objects as Files and Folders

Multi-threading + native C core + intelligent parsing → **FAST!**

Wide range of memory acquisition methods: **hardware** and **software**

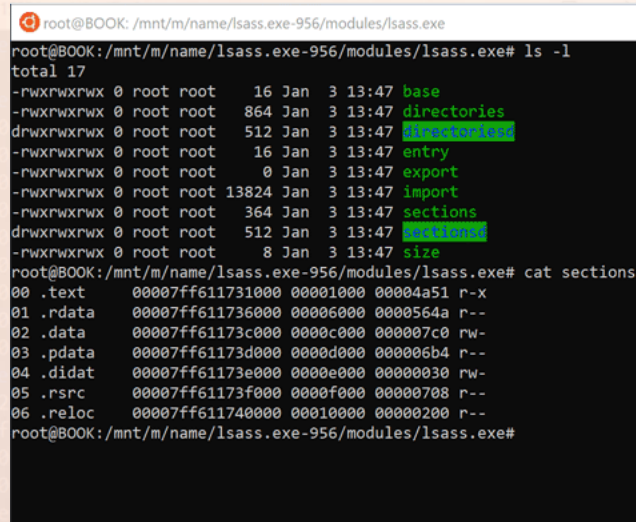


MemProcFS

DEMO: MOUNT 32GB IN A SECOND!

github.com/ufrisk/MemProcFS

Point and click memory analysis!



```

root@BOOK: /mnt/m/name/lsass.exe-956/modules/lsass.exe# ls -l
total 17
-rwxrwxrwx 0 root root 16 Jan 3 13:47 base
-rwxrwxrwx 0 root root 864 Jan 3 13:47 directories
drwxrwxrwx 0 root root 512 Jan 3 13:47 directories
-rwxrwxrwx 0 root root 16 Jan 3 13:47 entry
-rwxrwxrwx 0 root root 0 Jan 3 13:47 export
-rwxrwxrwx 0 root root 13824 Jan 3 13:47 import
-rwxrwxrwx 0 root root 364 Jan 3 13:47 sections
drwxrwxrwx 0 root root 512 Jan 3 13:47 sections
-rwxrwxrwx 0 root root 8 Jan 3 13:47 size
root@BOOK: /mnt/m/name/lsass.exe-956/modules/lsass.exe# cat sections
00 .text 00007ff611731000 00001000 00004a51 r-x
01 .rdata 00007ff611736000 00006000 0000564a r--
02 .data 00007ff61173c000 0000c000 00007c0b rw-
03 .pdata 00007ff61173d000 0000d000 00006b4a r--
04 .didat 00007ff61173e000 0000e000 00000030 rw-
05 .rsrsc 00007ff61173f000 0000f000 00000708 r--
06 .reloc 00007ff611740000 00010000 00000200 r--
root@BOOK: /mnt/m/name/lsass.exe-956/modules/lsass.exe#

```

MemProcFS - Design Goals

Ease of use – but yet powerful

Modular design and plugin functionality

APIs – C and Python

Performance



LeechCore

Memory Acquisition Library

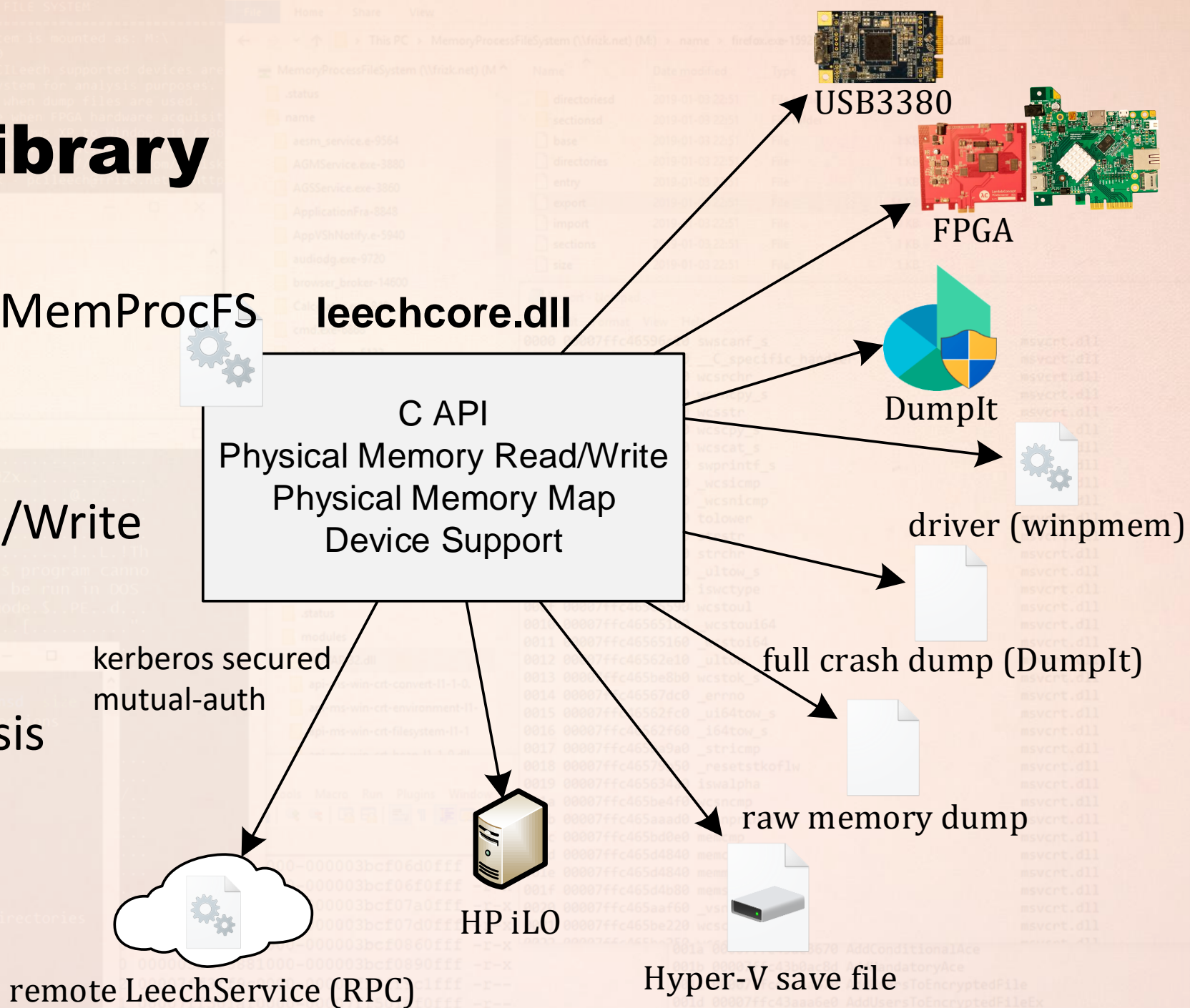
github.com/ufrisk/LeechCore

LeechCore Library

Used by PCILeech and MemProcFS

Focus:
Physical Memory Read/Write

Separates memory
acquisition from analysis





MemProcFS

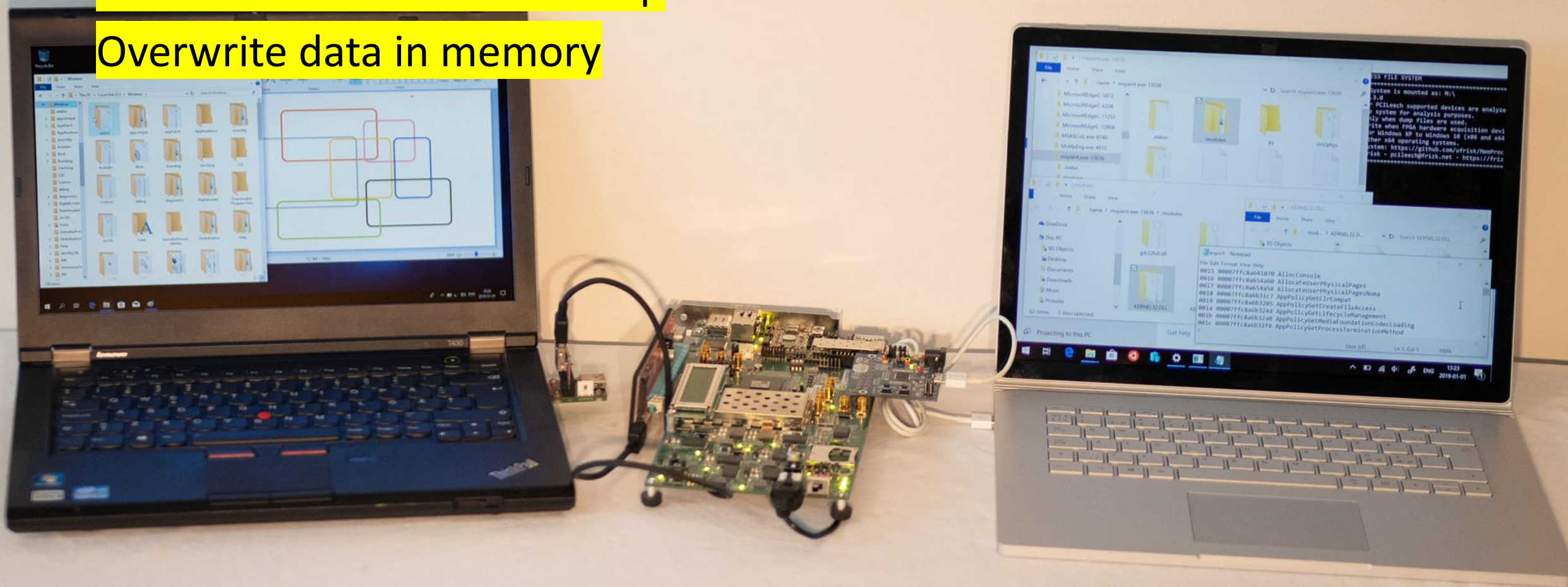
DEMO: LIVE MEMORY ANALYSIS AND ALTERATION

github.com/ufrisk/MemProcFS

Demo: Analysis and write to Live Memory

Hunt for data in Process Heap

Overwrite data in memory





MemProcFS

INCIDENT RESPONSE AND PYTHON API

github.com/ufrisk/MemProcFS

Incident Response with LeechAgent

Suspicious process → Computer Quarantined to VLAN

Limited bandwidth high latency network

Full memory dump == slow

Solution: Retrieve only the memory needed →

Analyze with The Memory Process File System

Or even better ... run the **analysis on** the **remote computer** by submitting a Python script!

Python API

Read / Write Physical and Virtual Memory

Process information

Modules information

List / Read / Write MemProcFS “files”

```
VmmPy_MemRead(  
VmmPy_MemReadScatter(  
VmmPy_MemWrite(  
VmmPy_MemVirt2Phys(  
VmmPy_PidList(  
VmmPy_PidGetFromName(  
VmmPy_ProcessGetMemoryMap(  
VmmPy_ProcessGetMemoryMapFromName(  
VmmPy_ProcessGetModuleMap(  
VmmPy_ProcessGetModuleFromName(  
VmmPy_ProcessGetInformation(  
VmmPy_ProcessListInformation(  
VmmPy_ProcessGetEAT(  
VmmPy_ProcessGetIAT(  
VmmPy_ProcessGetDirectories(  
VmmPy_ProcessGetSections(  
VmmPy_VfsList(  
VmmPy_VfsRead(  
VmmPy_VfsWrite(  
VmmPy_UtilFillHexAscii(  

```




MemProcFS

DEMO: REMOTE MALWARE MEMORY ANALYSIS

github.com/ufrisk/MemProcFS

Demo: Remote Malware Memory Analysis

Command Prompt

```
Q:\>MemProcFS.exe -device dumpit -remote rpc://kerberos-spn-remote-user:10.9.15.104
```

Analyze **live malware memory**

From **remote** infected system

By clicking on files and using API

The screenshot displays two windows used for remote malware memory analysis. The top window is 'map - Notepad', showing a list of files and their permissions. The bottom window is 'HxD - [M:\name\zuasi.exe-6700\vmem]', showing a hex dump of memory data.

map - Notepad

Offset	Size	Permissions	File Name
0016	2	00000000003fe000-00000000003fffff	-rw-
0017	3a	0000000000400000-0000000000439fff	-rwx zuasi.exe
0018	1	000000000043a000-000000000043afff	-r-x zuasi.exe
0019	1	0000000000440000-0000000000440fff	-rw-

HxD - [M:\name\zuasi.exe-6700\vmem]

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
000000401AC0	00	00	00	00	45	00	53	00	45	00	54	00	00	00	53	00	...E.S.E.T...S.
000000401AD0	65	00	63	00	75	00	72	00	69	00	74	00	79	00	00	00	e.c.u.r.i.t.y...
000000401AE0	00	00	00	00	45	00	53	00	45	00	54	00	00	00	41	00	...E.S.E.T...A.
000000401AF0	6E	00	74	00	69	00	76	00	69	00	72	00	75	00	73	00	n.t.i.v.i.r.u.s.
000000401B00	00	00	00	00	4D	00	69	00	63	00	72	00	6F	00	73	00	...M.i.c.r.o.s.
000000401B10	6F	00	66	00	74	00	00	00	49	00	6E	00	73	00	70	00	o.f.t...I.n.s.p.



PCILeech & MemProcFS

Hardware Cheats

github.com/ufrisk/MemProcFS

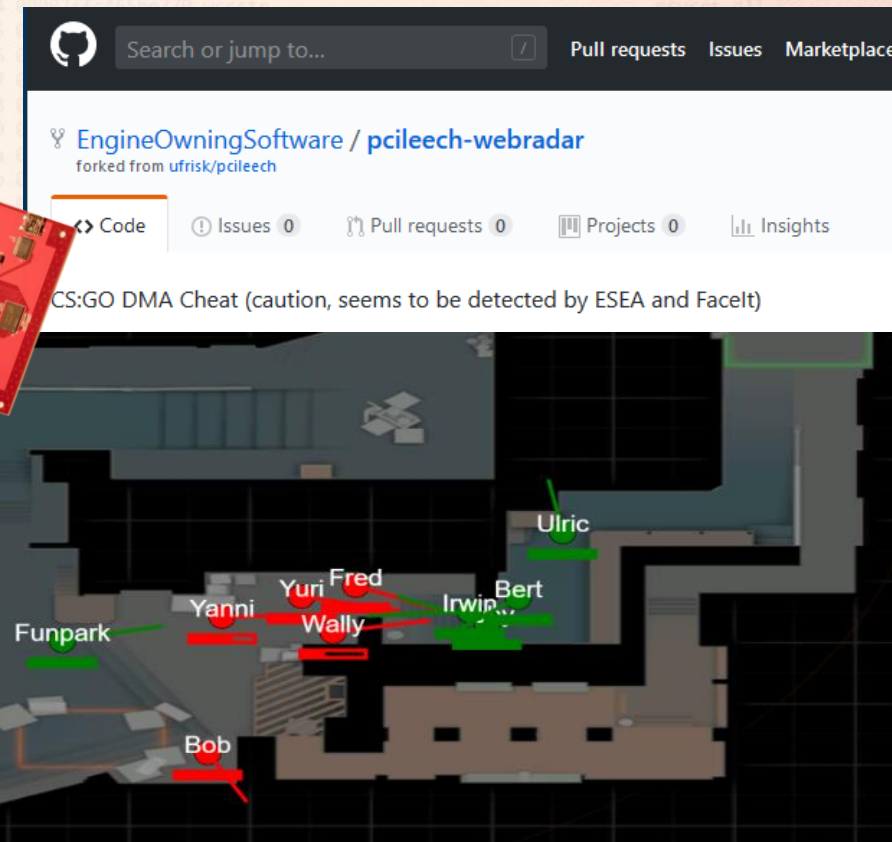
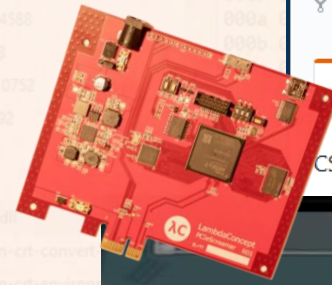
Hardware Cheats with PCILeech/MemProcFS

The **unexpected use case** – cheating in games!

Anti-Cheats – detects software based cheats

HW Cheat – “only” a PCIe device ...

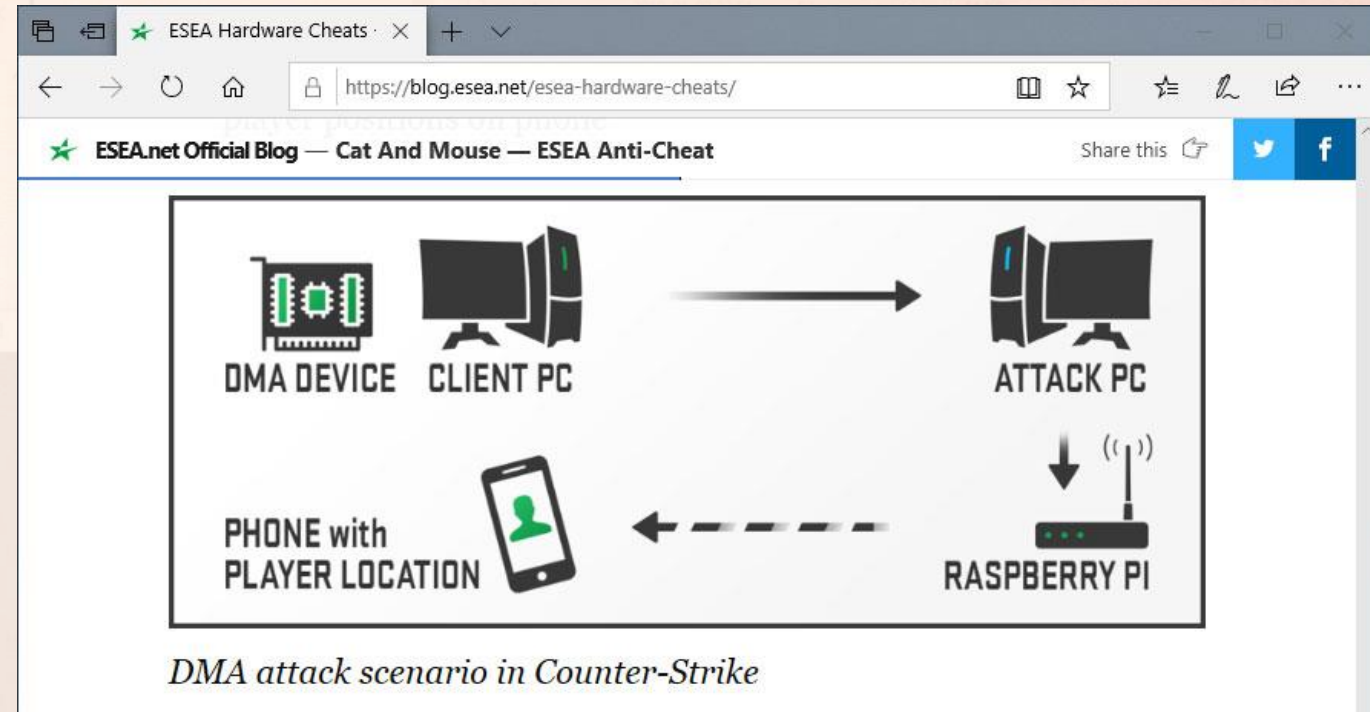
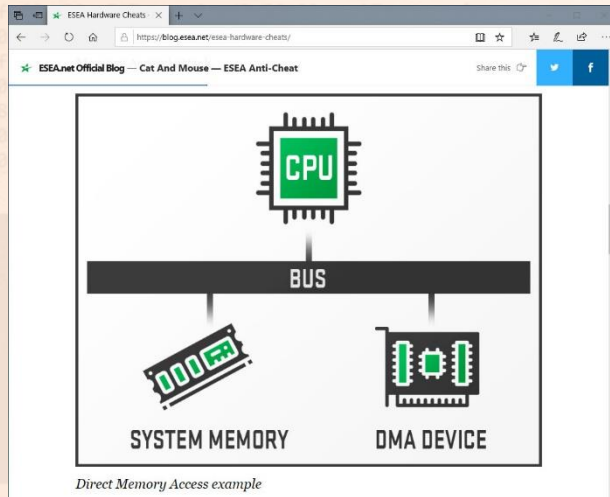
Read-Only “radar / map decloak”
or Read-Write (more easily detected)



Hardware Cheats



Hardware Cheats



“prices for these cheats have been seen in the **\$1,500 to \$5,000** range”

“ ... **ban wave** of both cheat customers and developers ...”

“ ... can detect hardware-based cheats even when disguising the hardware cheat as a legitimate device. ”

MemProcFS Summary

Fast – multi-threaded native C-core

Easy-to-Use – Point and click memory forensics in file system abstraction

Acquisition – Dump Files, Live Memory: driver, hw or remote agent!

Plugin support – roll your own plugins in C/C++/Python!

API – C/C++/Python

github.com/ufrisk/MemProcFS

Thank You!

Current Action: Dumping Memory
Access Mode: KMD (kernel module assisted DMA)
Progress: 8678 / 8678 (100%)
Speed: 173 MB/s
Address: 0x000000021E600000
Pages read: 2050967 / 2221568 (92%)
Pages failed: 170601 (7%)
Memory Dump: Successful.

 UlfFrisk

github.com/ulfrisk