SEC-T - 0x0Anniversary

Evil Devices and Direct Memory Attacks

UlfFrisk

Agenda

BUST FileVault open

PWN macOS, WINDOWS and **LINUX** by DMA code injection

DUMP memory at >150MB/s

MOUNT live file systems

Live **MEMORY FORENSICS**

EXECUTE code and **SPAWN SHELL**

About Me: Ulf Frisk

Penetration tester, online banking security

Employed in the financial sector – Stockholm, Sweden

MSc, Computer Science and Engineering

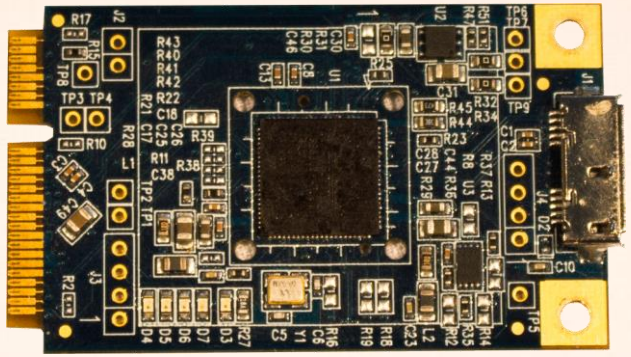
Previously presented at DEF CON 24, DEF CON 25, SEC-T 0x09

Interested in Low-Level programming and DMA

Disclaimer

This talk is given by me as an individual
My employer is not involved in any way

PCILeech == PLX USB3380 DEV BOARD + FIRMWARE + SOFTWARE

PCIe →  **← USB3**

$78 → $195 → Sold Out

No Drivers Required on Target!

No Assembly or Soldering Required!

>150MB/s DMA

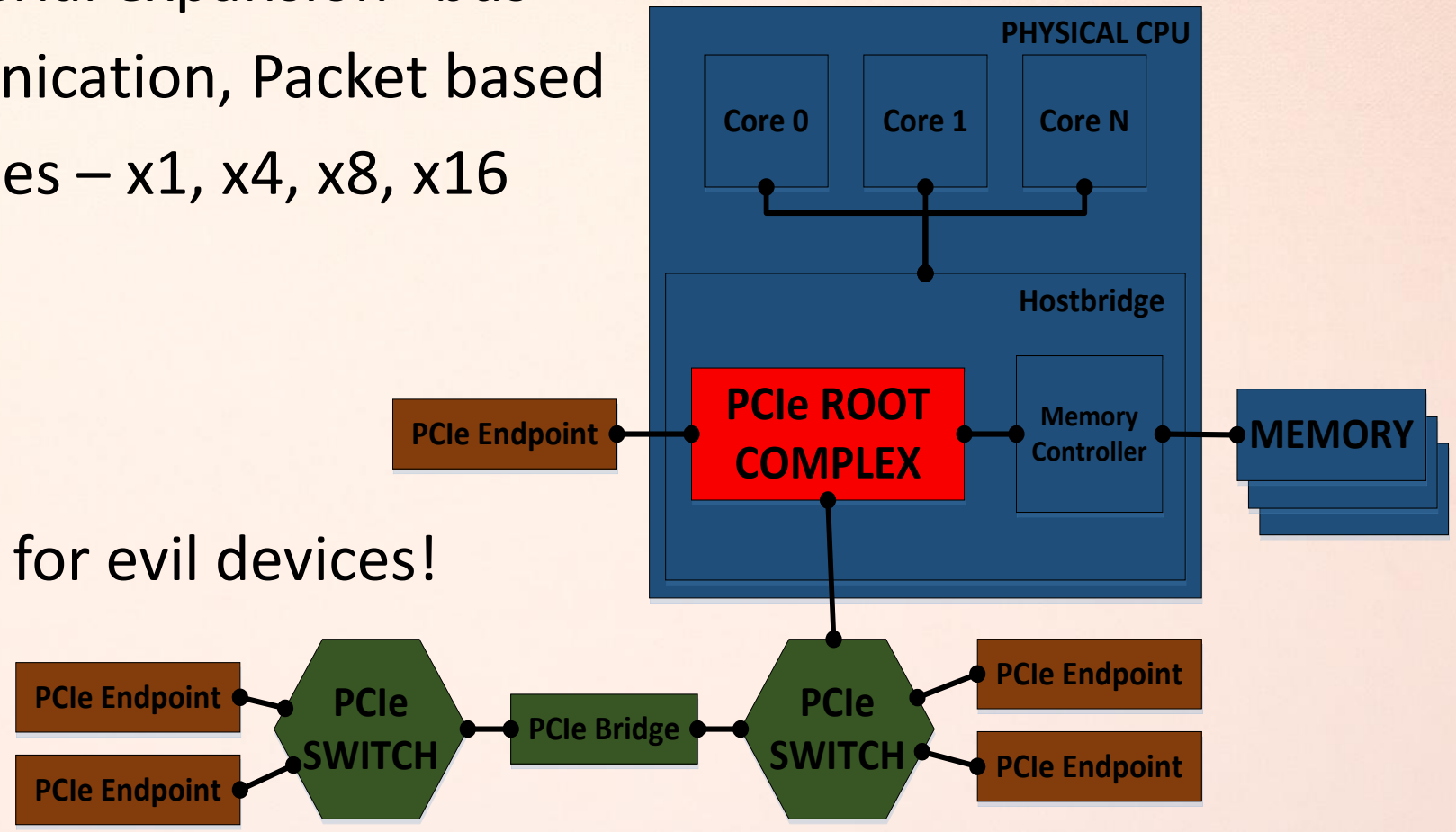
32-bit (<4GB) DMA only

Same hardware as the NSA Playset SLOTSCREAMER by @securelyfitz

PCI Express

- PCIe is a high-speed serial expansion ”bus”
- Point-to-point communication, Packet based
- From 1 to 16 serial lanes – x1, x4, x8, x16
- Hot pluggable
- DMA capable

- IOMMU == protection for evil devices!



PCI Express Form Factors

M.2 key B (+M)

M.2 key A+E

M.2 key M

Thunderbolt3 (USB-C)

Thunderbolt

PCIe x1

Mini PCIe

ExpressCard

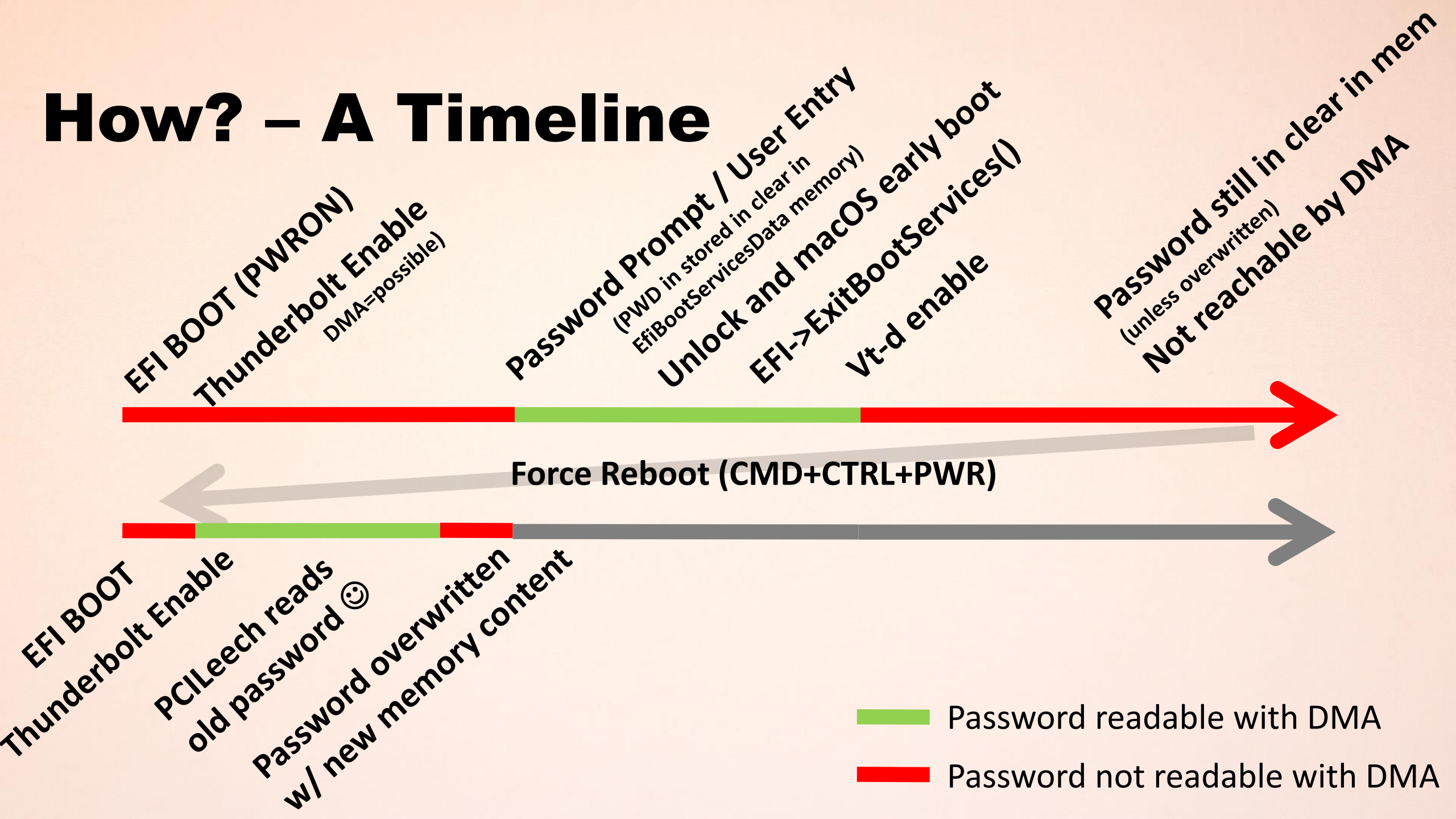
Everything here is PCI Express in different form factors and variations.

DEMO:

Bypass Filevault2 over Thunderbolt

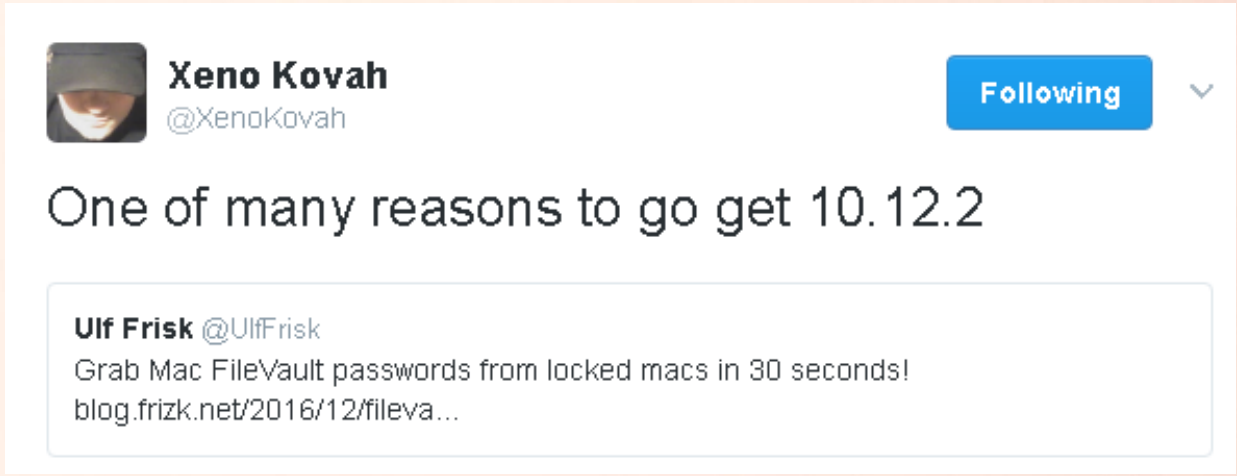
CVE-2016-7585
December 2016
March 2017

How? – A Timeline



The Patch

- Reported to Apple in August, **Patched December 13th**
- **Major EFI upgrade** co-bundled with macOS 10.12.2
- Stopping not only DMA/PCILeech but also OptionROMs and similar
- VT-d now enabled very early boot
- Fixes from APPLEs awesome firmware security team @XenoKovah, @coreykal



- Many **PC UEFIs** still not likely to be protected against DMA attacks …

It’s fixed – macs are now secure!

Fixed in all supported versions since March 2017

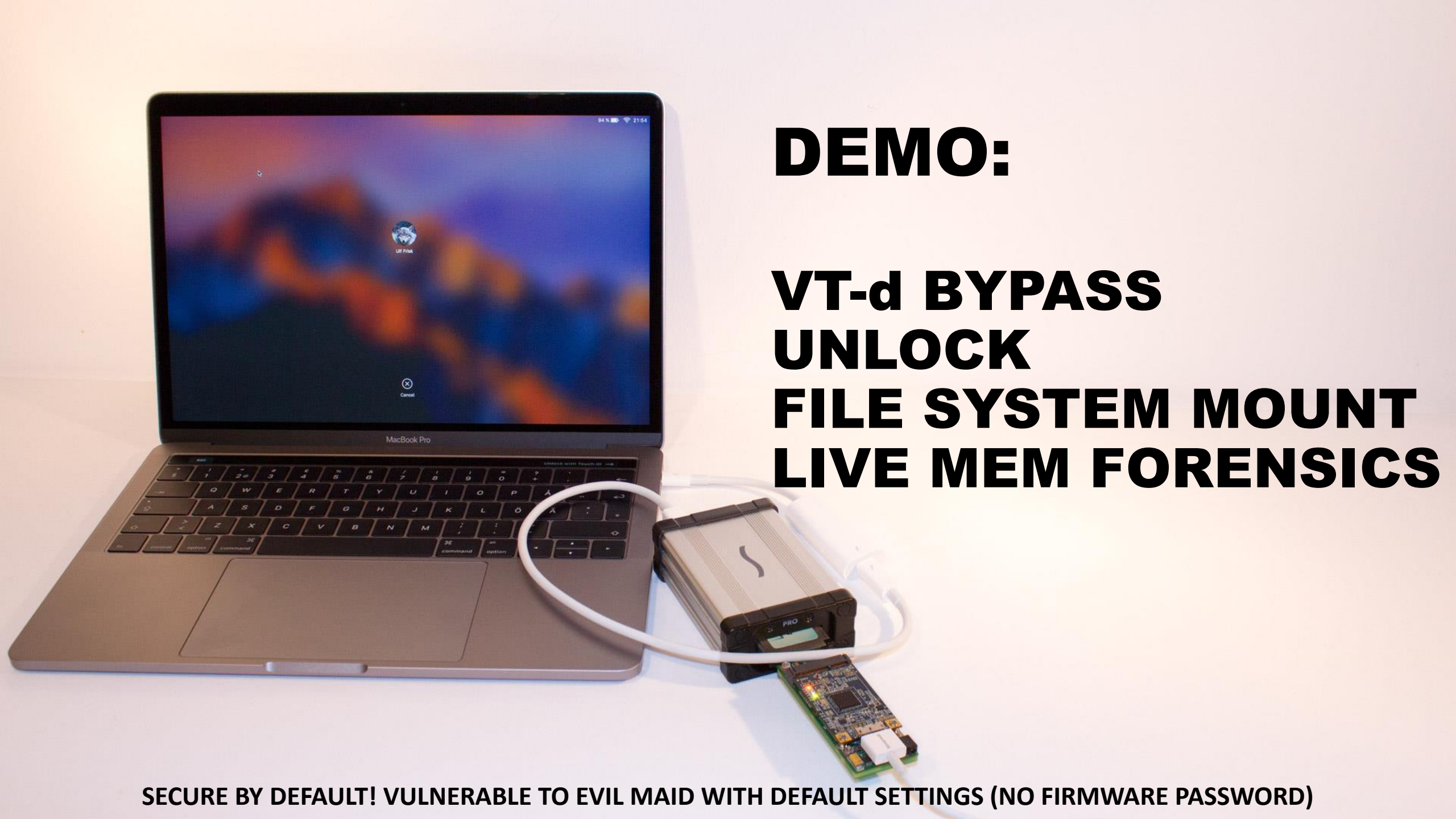
EFI

Available for: macOS Sierra 10.12.3

Impact: A malicious Thunderbolt adapter may be able to recover the FileVault 2 encryption password

Description: An issue existed in the handling of DMA. This issue was addressed by enabling VT-d in EFI.

CVE-2016-7585: Ulf Frisk (@UlfFrisk)



DEMO:

VT-d BYPASS
UNLOCK
FILE SYSTEM MOUNT
LIVE MEM FORENSICS

SECURE BY DEFAULT! VULNERABLE TO EVIL MAID WITH DEFAULT SETTINGS (NO FIRMWARE PASSWORD)

macOS

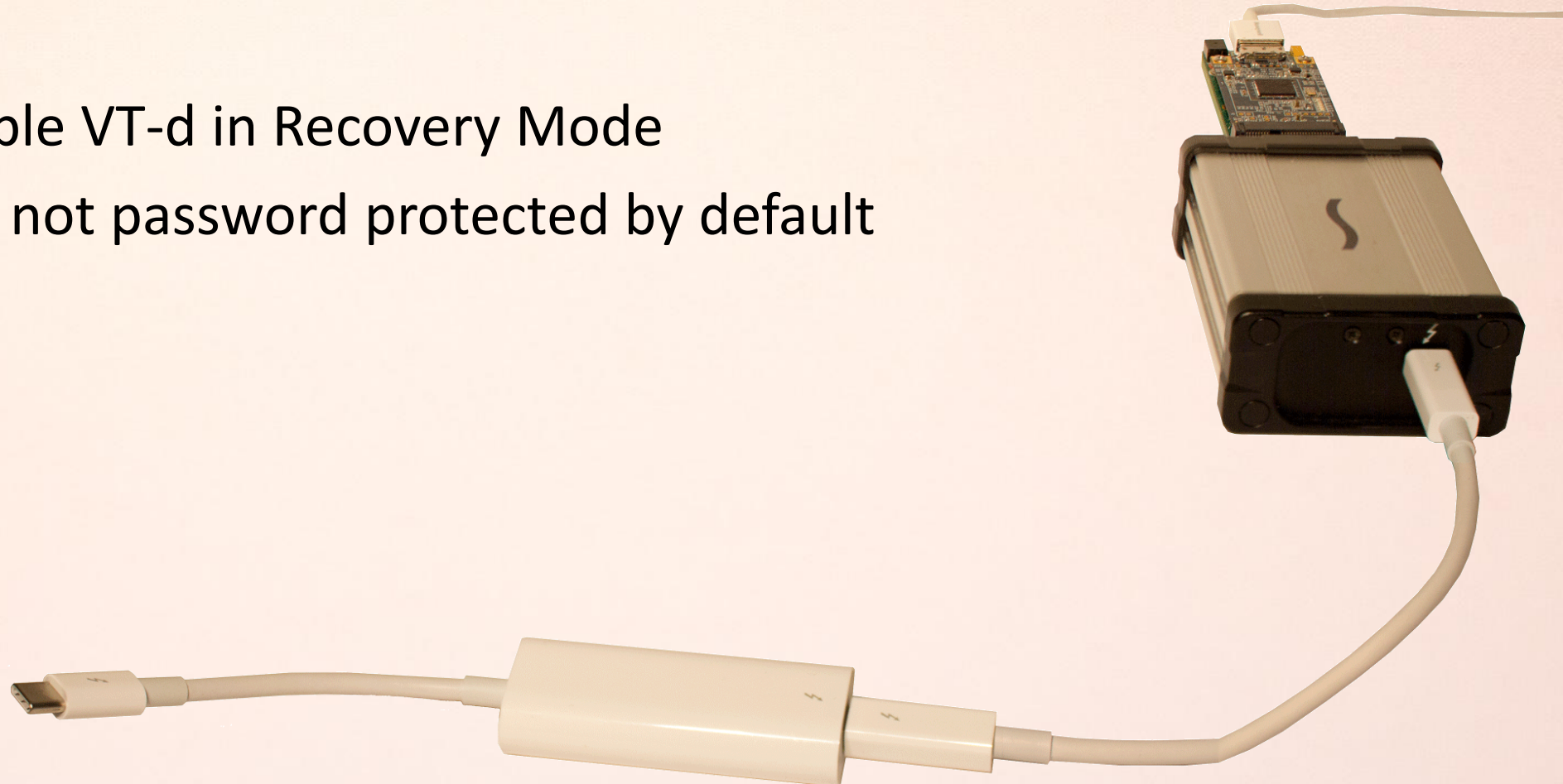
macOS

Thunderbolt and PCIe is protected with VT-d (IOMMU)

Possible to disable VT-d in Recovery Mode

Recovery Mode not password protected by default

→EVIL MAID!



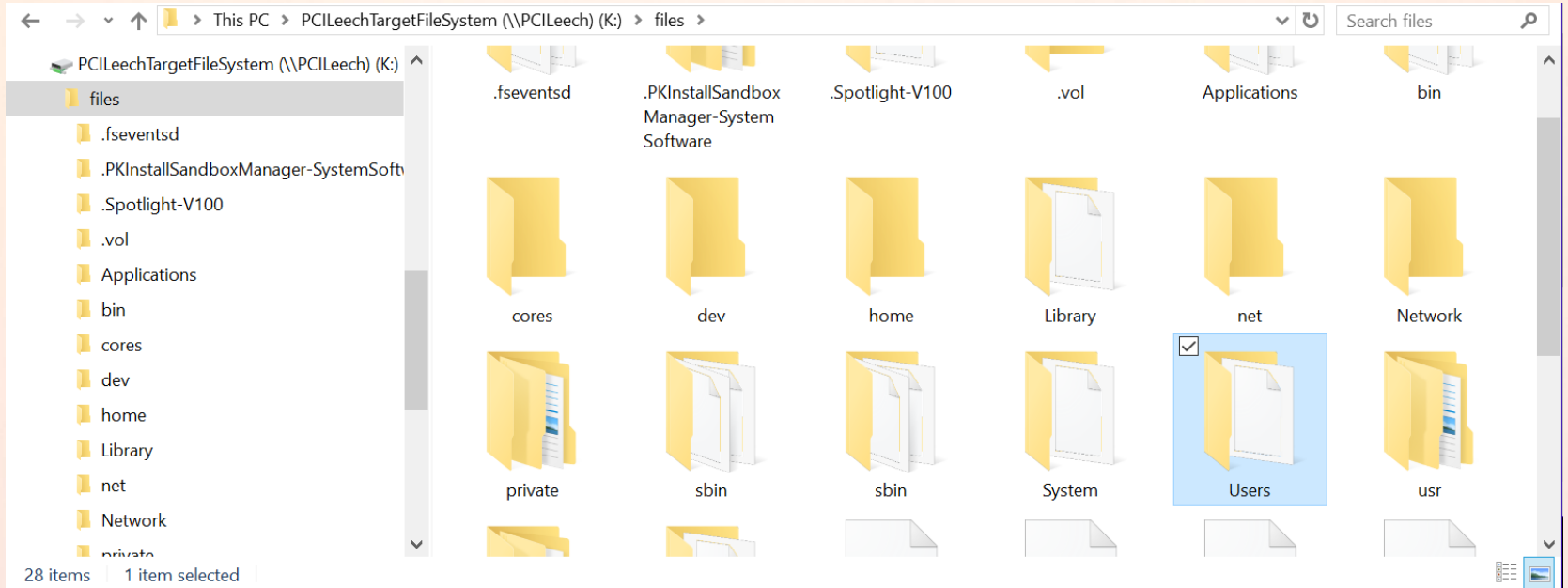
DMA Attacks made Super Easy!

Download **software**, Purchase **hardware**, flash, **PWN**!

Mount target file systems and memory as “network drive”

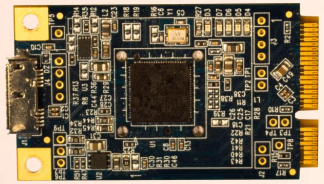
```
pcileech.exe mount -kmd macos
```

Click around, edit files on target – even raw disk devices and LIVE RAM!

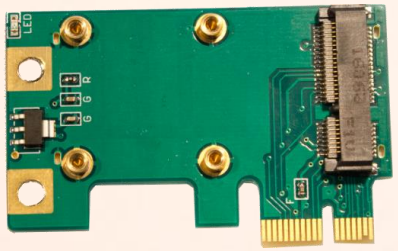


Adapters as building blocks

mini-PCIe
(internal/laptops)
(typically WiFi/4G)

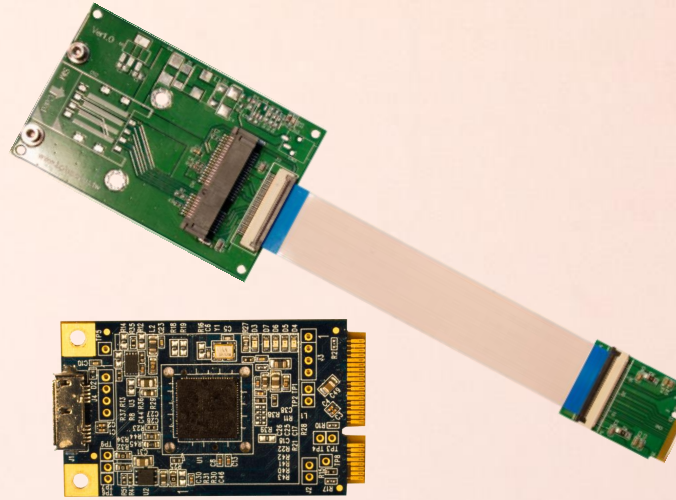


PCIe
(internal/desktops)



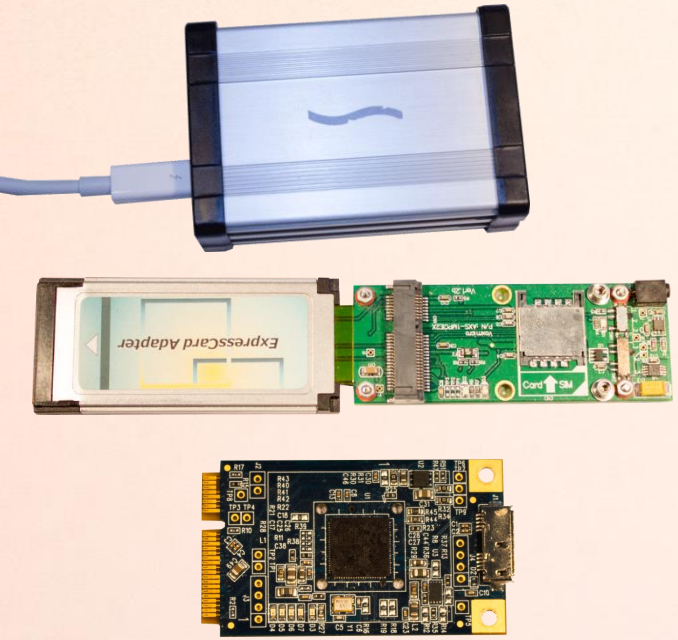
ExpressCard
(external/laptops)

M.2 Key A+E
(internal/laptops)
(typically WiFi/4G)

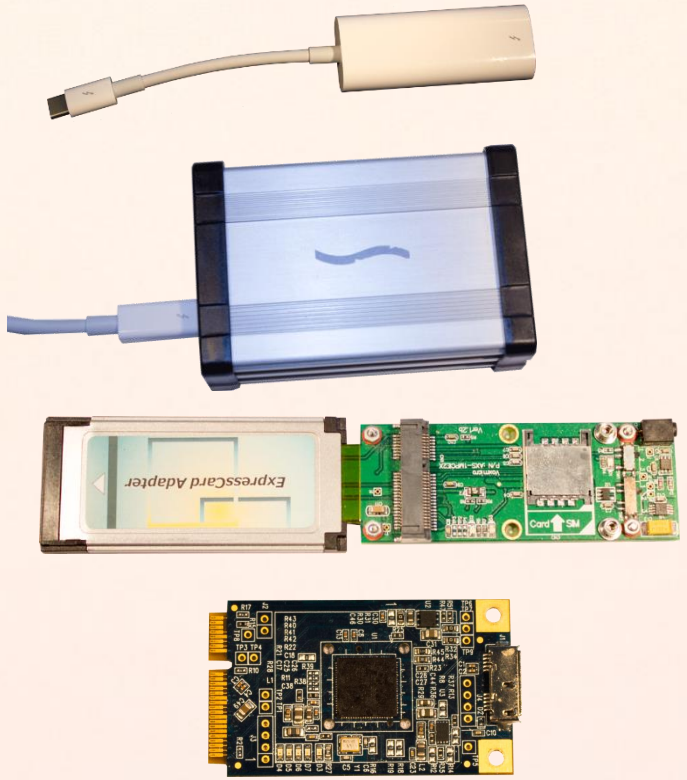


Adapters as building blocks

Thunderbolt



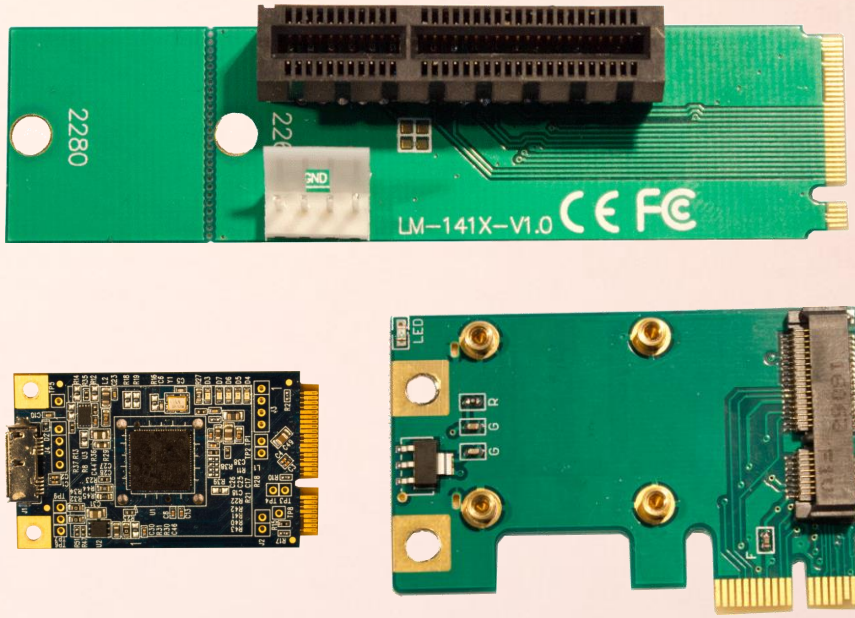
Thunderbolt3 (USB-C)



M.2 Key M

(internal/laptops)

(typically NVMe)



PWN all the operating systems

Most computers have more than 4GB memory!

Kernel Module can access all memory

Kernel Module can execute code

Search signatures using DMA and patch memory

Hijack kernel code execution

Windows

macOS

Windows 10

Kernel is located at top of memory

Kernel executable most often not directly reachable …

PAGE TABLES and **HAL** are located below 4GB ☺

Physical address 0x1000 often referenced as

HAL heap – It's the "low stub" containing the PROCESSOR_START_BLOCK struct

```
typedef struct _PROCESSOR_START_BLOCK {
    FAR_JMP_16 Jmp;
    ULONG CompletionFlag;
    PSEUDO_DESCRIPTOR_32 Gdt32;
    PSEUDO_DESCRIPTOR_32 Idt32;
    KGDTENTRY64 Gdt[PSB_GDT32_MAX + 1];
    ULONG64 TiledCr3;
    FAR_TARGET_32 PmTarget;
    FAR_TARGET_32 LmIdentityTarget;
    PVOID LmTarget;
    PPROCESSOR_START_BLOCK SelfMap;
    ULONG64 MsrPat;
    ULONG64 MsrEFER;
    KPROCESSOR_STATE ProcessorState
} PROCESSOR_START_BLOCK;
```

The magical 0x1000 address

Kernel paging base (**CR3/PML4**) address located at **0x10a0**

Virtual address of HAL “HEAP” located at **0x1078**

Search HAL memory for function pointer table

Overwrite **hal!HalpApicRequestInterrupt** pointer with PCILeech function pointer

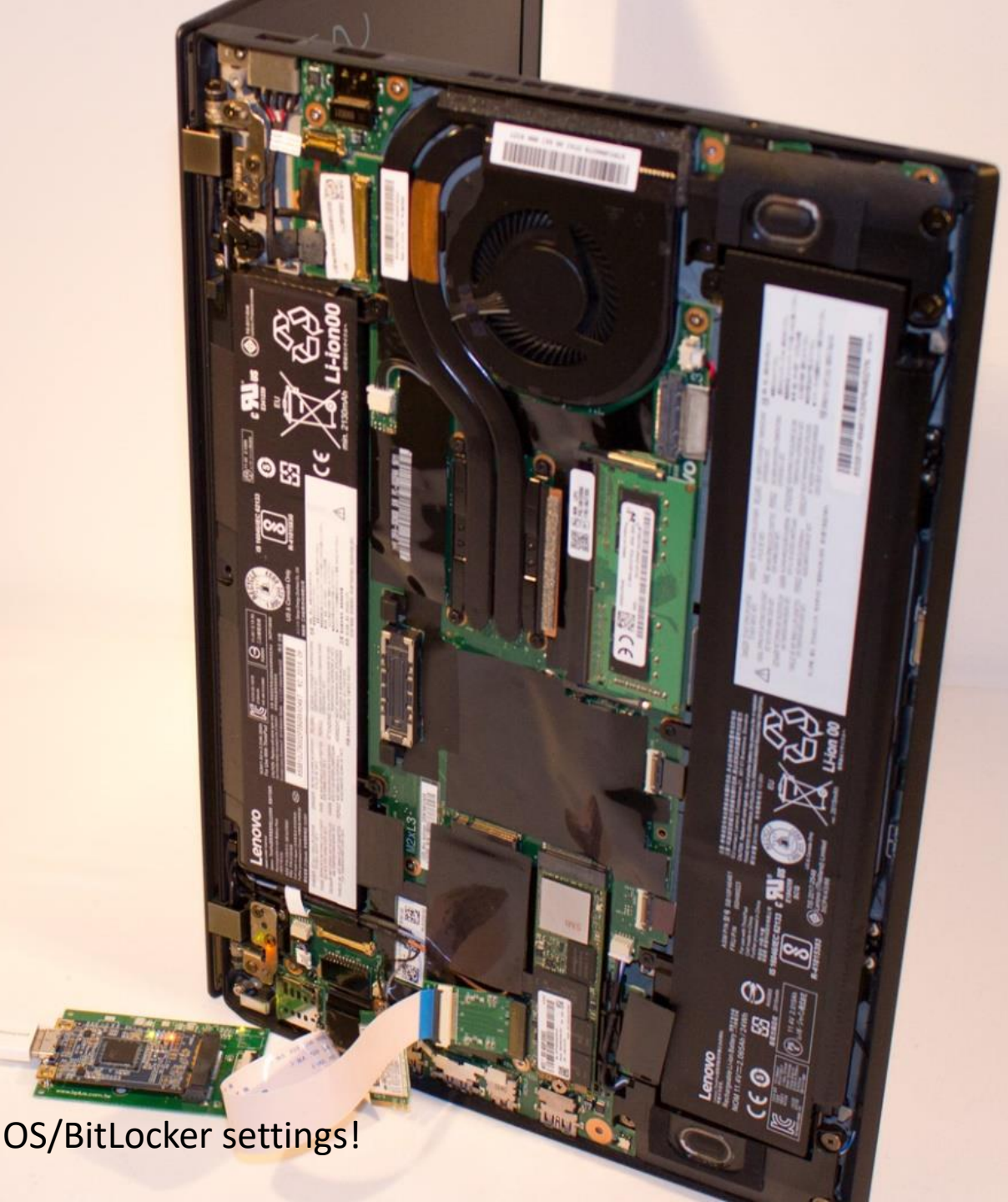
SUCCESS :)

```
Q:\>pcileech pagedisplay -min 0x1000

Memory Page Read: Page contents for address: 0x0000000000001000
0000    e9 4d 06 00 01 00 00 00  01 00 00 00 3f 00 18 10   .M..........?...
0010    00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
0020    00 00 00 00 00 00 00 00  00 00 00 00 00 9b 20 00   .............. .
0030    00 00 00 00 00 00 00 00  ff ff 00 00 00 93 cf 00   ................
0040    00 00 00 00 00 00 00 00  ff ff 00 00 00 9b cf 00   ................
0050    00 00 00 00 00 00 00 00  00 80 a0 d6 00 00 00 00   ................
0060    7c 16 00 00 30 00 c1 16  00 00 10 00 00 00 00 00   |...0...........
0070    40 a8 22 69 03 f8 ff ff  00 80 00 40 9e f7 ff ff   @."i.......@....
0080    06 01 07 00 06 01 07 00  01 09 00 00 00 00 00 00   ................
0090    33 00 05 80 00 00 00 00  00 00 00 00 00 00 00 00   3...............
00a0    00 80 1b 00 00 00 00 00  f8 06 15 00 00 00 00 00   ................
00b0    00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
```

DEMO:

Kernel Implant
DUMP MEMORY
SPAWN SHELL
UNLOCK

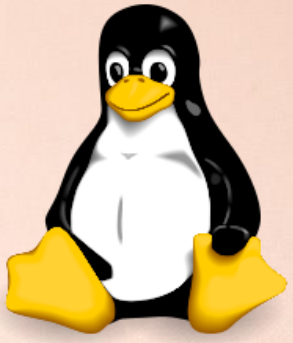


WIN10 is not secure by default, but can be secured
with Virtualization Based Security (VBS) and correct BIOS/BitLocker settings!

ALSO POSSIBLE:
ATTACK FROM PHONE
PWN WIN7

WINDOWS 7 IS NOT SECURE! AND IS NOT LIKELY TO BE SECURED...

Linux



KASLR randomizes kernel location fully in physical memory

VT-d often not used by default

Cannot reach kernel directly on most computers (>4GB RAM)

Patch UEFI Runtime Services to gain code execution

Pwn the kernel :)

EFI Runtime Services

Part of UEFI that lingers after OS is started

Provides functionality like:

- Get/Set Time, Variable
- ResetSystem/UpdateCapsule
- …

Always located < 4GB

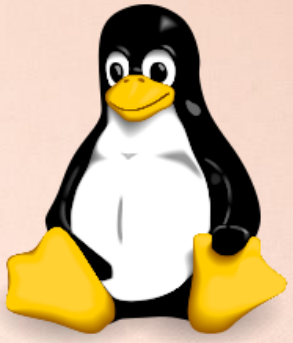
Dispatch table RUNTSERV with function pointers!

Known signature

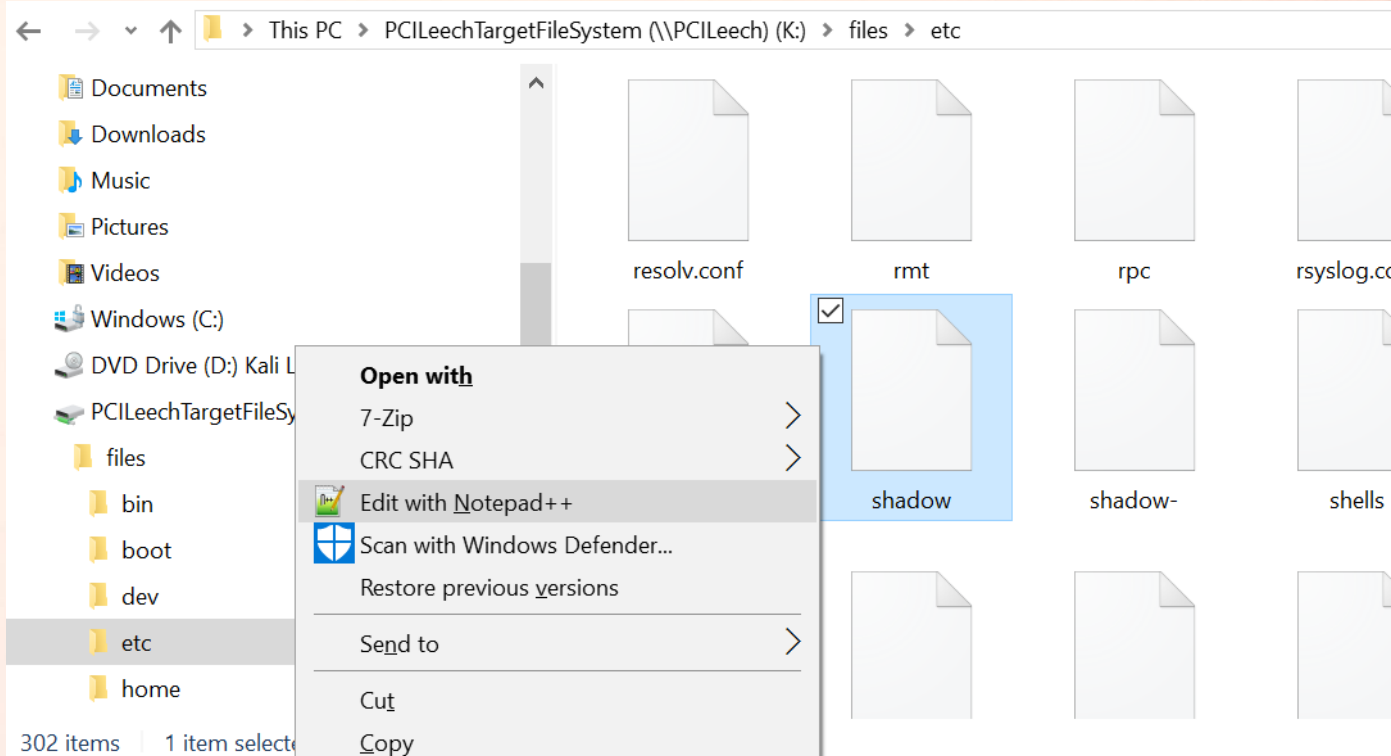
```
Q:\>pcileech.exe pagedisplay -min 0xda659e60

Memory Page Read: Page contents for address: 0x00000000DA659000
0000    70 68 64 30 d8 00 00 00  06 00 00 00 00 00 00 00   phd0............
0010    38 31 0c d0 00 00 00 00  65 76 6e 74 00 00 00 00   81......evnt....
....
0e00    70 68 64 30 a8 00 00 00  06 00 00 00 00 00 00 00   phd0............
0e10    38 31 0c d0 00 00 00 00  52 55 4e 54 53 45 52 56   81......RUNTSERV
0e20    1f 00 02 00 88 00 00 00  9e ef 55 1f 00 00 00 00   ..........U.....
0e30    d8 8a 1f fc fe ff ff ff  60 8c 1f fc fe ff ff ff   ........`.......
0e40    c8 8d 1f fc fe ff ff ff  b4 8f 1f fc fe ff ff ff   ................
0e50    8c 84 c0 d6 00 00 00 00  b0 83 c0 d6 00 00 00 00   ................
0e60    28 bb 14 fc fe ff ff ff  d4 bc 14 fc fe ff ff ff   (...............
0e70    30 be 14 fc fe ff ff ff  f8 c2 1f fc fe ff ff ff   0...............
0e80    d0 85 1d fc fe ff ff ff  78 d3 1f fc fe ff ff ff   ........x.......
0e90    30 d5 1f fc fe ff ff ff  bc bf 14 fc fe ff ff ff   0...............
0ea0    70 74 61 6c a8 00 00 00  c9 ef 76 e4 f5 c8 7a d9   ptal......v...z.
```

EFI Runtime Services Hijack

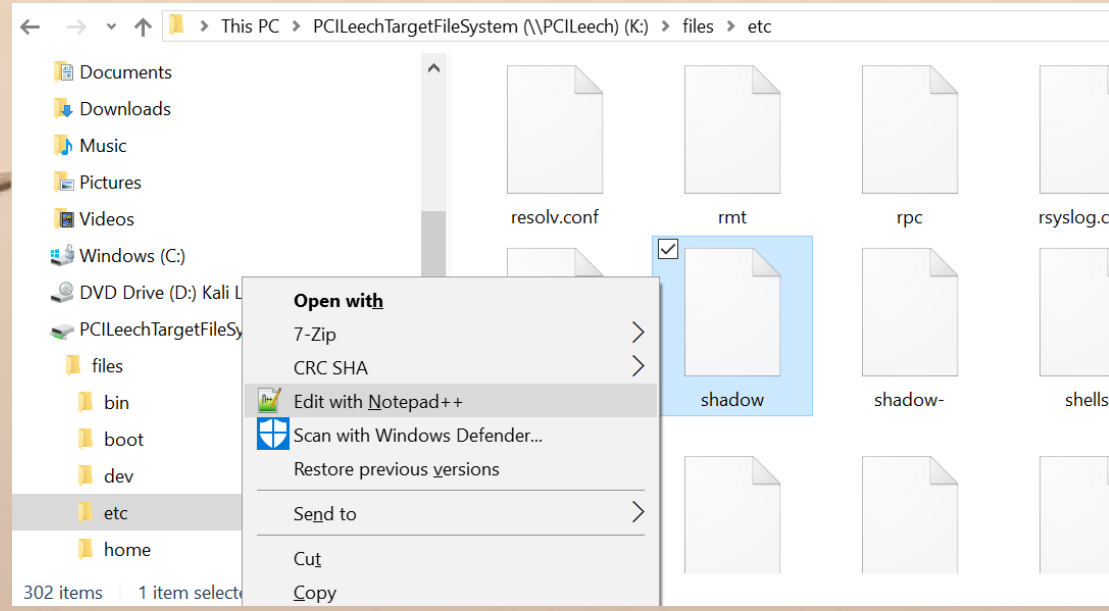


1. Locate RUNTSERV table
2. Patch function pointers
3. Wait for runtime services code execution
4. Patch Linux kernel (>4GB), Restore Runtime Services, Resume normal execution.
5. Wait for kernel execution
6. SUCCESS :)



DEMO:

MOUNT file system
UNLOCK (edit /etc/shadow)



LINUX IS SECURE/INSECURE DEPENDING ON CONFIGURATION AND DISTRIBUTION ...

Mitigations

Keep macOS up-to-date and set **firmware password**

Hardware without DMA ports

BIOS password, DMA port lockdown, Pre-boot authentication

IOMMU / VT-d (Virtualization Based Security, DMA Guard [rs3], …)

Use Cases

Awareness – full disk encryption is not invincible …

Pentesting and **RedTeaming**

Excellent for **forensics**

PLEASE DO NOT DO EVIL with this tool

Key Takeaways

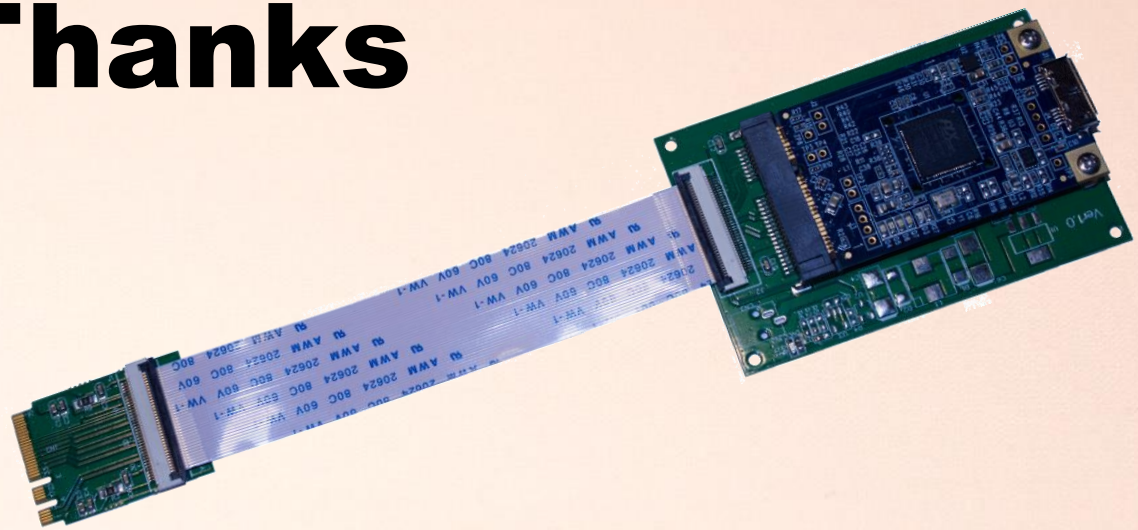
PHYSICAL ACCESS is still an issue

- be aware of potential **EVIL MAID** attacks

PCILeech == DMA Attacks made **Super Easy!**

USED for GOOD and **EVIL**

References and Thanks



PCILeech

github.com/ufrisk/pcileech

twitter.com/UlfFrisk

blog.frizk.net

SLOTSCREAMER

www.nsaplayset.org/slotscreamer by Joe Fitzpatrick and Miles Crabill

Previous Work and Thanks to

Inception Firewire DMA: Carsten Maartmann-Moe

The “low stub” and PCILeech contributions: Alex Ionescu

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.
```

github.com/ufrisk/pcileech

UlfFrisk