



TWO SIGMA



LEGBACORE



# Thunderstrike 2: Sith Strike

Trammell Hudson – Two Sigma

Xeno Kovah, Corey Kallenberg – LebgaCore

# About us – Trammell Hudson

- I like to take things apart.
  - Magic Lantern: firmware for Canon DSLR cameras
  - Ghosts in the ROMs: Easter Eggs in old devices
  - Thunderstrike: First firmware rootkit for Macs
- Two Sigma is a high tech investment management firm and we're concerned about the {software,firmware,hardware,etc} security of our systems.

# About us - LegbaCore

- We do digital voodoo
- Newly independent as of January 2015
- The only company focused primarily on x86 firmware & peripheral firmware (HD/NIC/GPU/EC/ME) security

# This talk has 1 main point

- Apple has not been as responsive, or as accurate, as other PC vendors in responding to industry-wide notifications of firmware vulnerabilities. Consequently Mac users have been left vulnerable to attacks that have been fixed on other x86-based PCs.

# Everything old is new again (to Apple)!

Vulnerability	Private CERT/CC (and USRT once created) Disclosure Date	Public Disclosure Date	Macs Vulnerable as of Conference Submission?	Patches available as of conference date?
VU #976132 (Darth Venamis)	Sept. 2014	Dec. 2014	Yes	Partial fix released 6/30/2015
VU #577140 (Snorlax & Prince Harming)	Aug. 2013 & N/A	May 2015	Yes	Yes – released 6/30/2015
VU# 766164 (Speed Racer)	May 2014	Dec. 2014	Yes	No
VU #255726 (The Sicilian)	Approx. May 2013 (we don't have the emails anymore)	Sept. 2013	Some Yes, still investigating	No
VU# 758382 Setup UEFI Variable Bug	Intel - ? Us – Sept. 2013	Mar. 2014	No (they don't have this UEFI variable)	N/A

# Case study: Speed Racer

CERT VU#766164

- Because it's a *hardware race condition!* Get it?!



# Case study: Speed Racer

## CERT VU#766164

### Vendor Information [\(Learn More\)](#)

(Picture retrieved Jun 12<sup>th</sup> 2015)

Vendor	Status	Date Notified	Date Updated
American Megatrends Incorporated (AMI)	Affected	12 Sep 2014	29 Dec 2014
Phoenix Technologies Ltd.	Affected	12 Sep 2014	17 Dec 2014
Apple Inc.	Not Affected		16 Dec 2014
Dell Computer Corporation, Inc.	Not Affected	12 Sep 2014	21 Jan 2015
IBM Corporation	Not Affected	12 Sep 2014	16 Dec 2014
Insyde Software Corporation	Not Affected	12 Sep 2014	03 Feb 2015
Intel Corporation	Not Affected	12 Sep 2014	06 Jan 2015
AsusTek Computer Inc.	Unknown	12 Sep 2014	12 Sep 2014
Gateway	Unknown	12 Sep 2014	12 Sep 2014
Hewlett-Packard Company	Unknown		If you don't hold your vendor accountable: silence
Lenovo	Unknown		
Sony Corporation	Unknown	12 Sep 2014	12 Sep 2014
Toshiba	Unknown	12 Sep 2014	12 Sep 2014

# Case study: Speed Racer

CERT VU#766164

- Sam Cornwell, John Butterworth, and Rafal Wojtczuk all independently realized the potential for a race condition in the BIOS\_CNTL.BLE (BIOS Lock Enable) protection bit. Corey then actually proved it with a PoC
- Disclosed to Intel & CERT/CC in May 2014
- Disclosed publicly at 31C3 in Dec. 2014

# Case study: Speed Racer

CERT VU#766164

- Hardware race condition in the interaction between the CPU & flash programming hardware
- The oldest BIOS protection bit, BIOS\_CNTL.BLE (BIOS Lock Enable), can *always* be bypassed on Intel systems
- Intel added a new bit, BIOS\_CNTL.SMM\_BWP that fixes this, but BIOS makers need to design for it and enable it
- Protected Range Registers (PRRs) also prevent writing to the BIOS, even if the race is successful

# MacMini7,1 (newest available) Protections

OS-resident Attacker

**BIOS\_CNTL.  
BLE bit**

ACCESS DENIED TO NON-SMM

**BIOS\_CNTL.  
SMM\_BWP bit**

ACCESS DENIED TO NON-SMM

**Protected  
Range  
Registers**

ACCESS DENIED  
EVEN TO SMM

ACCESS DENIED  
EVEN TO SMM

Firmware

UEFITool  
says  
“padding”?

Code & Stuff

EFI Variables

Code & Stuff

Flash  
Addr.

18E000

190000

610000

632000

10

7FFFFF

# MacMini7,1 (newest available) Protections

OS-resident Attacker

**BIOS\_CNTL**  
**BLE bit**

ACCESS DENIED TO NON-SMM

**BIOS\_CNTL**  
**SMM\_BWP bit**

LOL! J/K  
APPLE DOESN'T USE THIS!

**Protected  
Range  
Registers**

ACCESS DENIED  
EVEN TO SMM

ACCESS DENIED  
EVEN TO SMM

Firmware

UEFITool  
says  
“padding”?

Code & Stuff

EFI Variables

Code & Stuff

Flash  
Addr.

18E000

190000

610000

632000

11

7FFFFF

# MacMini7,1 (newest available) Protections

OS-resident Attacker

BIOS\_CNTL  
BLE bit

LOL! J/K APPLE DOESN'T USE THIS EITHER!

*But if they DID try to use it...*

*w/o SMM\_BWP it would be architecturally vulnerable to Speed Racer :)*

BIOS\_CNTL  
SMM\_BWP bit

Protected  
Range  
Registers

ACCESS DENIED  
EVEN TO SMM

ACCESS DENIED  
EVEN TO SMM

Firmware

UEFITool  
says  
“padding”?

Code & Stuff

EFI Variables

Code & Stuff

Flash  
Addr.

18E000

190000

610000

632000

12

7FFFFF

# MacMini7,1 (newest available) Protections

OS-resident Attacker

**BIOS\_CNTL**  
**BLE bit**

**BIOS\_CNTL**  
**SMM\_BWP bit**

**Protected  
Range  
Registers**



ACCESS DENIED  
EVEN TO SMM

ACCESS DENIED  
EVEN TO SMM

Firmware

UEFITool  
says  
“padding”?

Code & Stuff

EFI Variables

Code & Stuff

Flash  
Addr.

18E000

190000

610000

632000

13

7FFFFF

# So what happens when we write to the gaps?

- <https://youtu.be/kOrj323ddOQ>

# Case study: Speed Racer

CERT VU#766164

- We haven't seen any Macs that follow the Intel's best practices and use BIOS\_CNTL.SMM\_BWP
- Apple said they're “not affected” by Speed Racer...And in a *strict* sense, that's true... because there's no need to utilize an exploit to bypass a protection mechanism when the vendor doesn't even use the protection was never available to begin with!

# Patch Status: Unpatched/ Architecturally vulnerable

- If Apple just used the BIOS\_CNTL.SMM\_BWP bit (like other vendors), it could make it so that you need to first break into SMM before you can smash stuff and brick the system



# Case study: Darth Venamis

CERT VU#976132

- (Sometimes mis-referred to as the “Dark Jedi” attack)
- So named by Rafal Wojtczuk because Darth Plagueis defeated Darth Venamis, and put him into a death-like sleep/coma to study midi-chlorians



Venamis

Plagueis

# Case study: Darth Venamis

CERT VU#976132

- When an Intel system goes into a low-power mode, such as ACPI S3 sleep, *some of the BIOS protection bits become unlocked* the same way they would if the system is restarting
- Disclosed to CERT/CC and UEFI Security Response Team (USRT) in Sept. 2014
- Made public at 31C3 in Dec. 2014 [6][8]

# Case study: Darth Venamis

## CERT VU#976132

- For some reason, with this VU# CERT stopped listing the vendors they had reached out to that said they weren't vulnerable, and those that they reached out to that didn't respond
- That's the whole reason we liked using CERT! :-)

### Vendor Information [\(Learn More\)](#)

(Picture retrieved Jun 12<sup>th</sup> 2015)

Vendor	Status	Date Notified	Date Updated
American Megatrends Incorporated (AMI)	Affected	15 Sep 2014	10 Dec 2014
Dell Computer Corporation, Inc.	Affected	15 Sep 2014	22 Jan 2015
Insyde Software Corporation	Affected	-	03 Feb 2015
Intel Corporation	Affected	15 Sep 2014	29 Dec 2014
Lenovo	Affected	-	21 Jan 2015
Phoenix Technologies Ltd.	Affected	06 Oct 2014	19 Dec 2014

# Case study: Darth Venamis

CERT VU#976132

- So we don't know whether
  - CERT didn't contact them
  - Apple didn't respond
  - Apple responded and said they were not vulnerable
- We asked CERT which was the case, and CERT never replied

# The Venamis Effect

OS-resident Attacker

**BIOS\_CNTL**  
**BLE bit**

**BIOS\_CNTL**  
**SMM\_BWP bit**

**Protected  
Range  
Registers**

Goodbye cruel world!

Goodbye  
cruel world!

Firmware

UEFITool  
says  
“padding”?

Code & Stuff

EFI Variables

Code & Stuff

Flash  
Addr.

18E000

190000

610000

632000

21

7FFFFF

# The Venamis Effect

OS-resident Attacker

**BIOS\_CNTL**  
**BLE bit**

**BIOS\_CNTL**  
**SMM\_BWP bit**

**Protected  
Range  
Registers**

Firmware	UEFITool says “padding”?	Code & Stuff	EFI Variables	Code & Stuff
Flash Addr.	18E000	190000	610000	632000
				22 7FFFFF

# #PrinceHarming



See the whitepaper for why Snorlax is here ;)

# #PrinceHarming

- Pedro Vilaca (@osxreverser) saw the abstract for this talk. He correctly inferred based on the title that it would have to do with Macs' containing the Venamis vulnerability
- May 29th 2015, Pedro posted[14] that while he was investigating Venamis, he found an easier attack: Just put the system to sleep, and the firmware protection registers *are unlocked on wake!*
  - Named #PrinceHarming by Katie Moussouris (@k8em0)
  - We have seen this problem before, in CERT VU#[577140](#), but hadn't actually seen it yet on Macs. Because it turned out Xeno's MBP11,2 wasn't vulnerable (even though it was vulnerable to Venamis.)
- So basically Pedro accidentally dropped a Mac firmware vulnerability Oday :)
  - The *real* question is, why are there trivial-to-exploit Apple firmware Odays just laying around out in the open?

# Patch Status

- On June 30<sup>th</sup> 2015 Apple released an update[19] for 24 models to patch Venamis & Prince Harming (basically for everything since 2011)
  - iMac (27-inch, Mid 2011) ("iMac12,1") until latest
  - ("iMac15,1") Mac Mini (Mid 2011) with Intel graphics ("Macmini5,1") until latest ("Macmini7,1")
  - MacBook Air (11-inch, Mid 2011) ("MacBookAir4,1") until latest ("MacBookAir7,1")
  - MacBook Pro (13-inch, Late 2011) ("MacBookPro8,1") until latest ("MacBookPro12,1")
  - MacBook (Early 2015) ("MacBook8,1")
  - Mac Pro (Late 2013) ("MacPro6,1")

# What wasn't patched

- Security update 2015-001 sets only sets the PRRs and locks FLOCKDN in PEI before the S3 bootscript is interpreted
- Therefore an attacker can still use Venamis to infect SMM
  - E.g. attacker can still bypass SMM\_BWP if Apple ever bothers to use this protection mechanism
- We informed Apple of this additional issue prior to the release of the update

# Reminder: This talk has 1 main point

- Apple has not been as responsive, or as accurate, as other PC vendors in responding to industry-wide notifications of firmware vulnerabilities. Consequently Mac users have been left vulnerable to attacks that have been fixed on other x86-based PCs.

A dramatic, close-up shot of a shark's head, focusing on its open mouth and sharp, yellowish-white teeth. The shark's skin is dark and textured. Its glowing yellow eyes are visible above the mouth. The background is dark and out of focus.

ENOUGH!  
GET TO  
THE REAL  
ATTACKS!

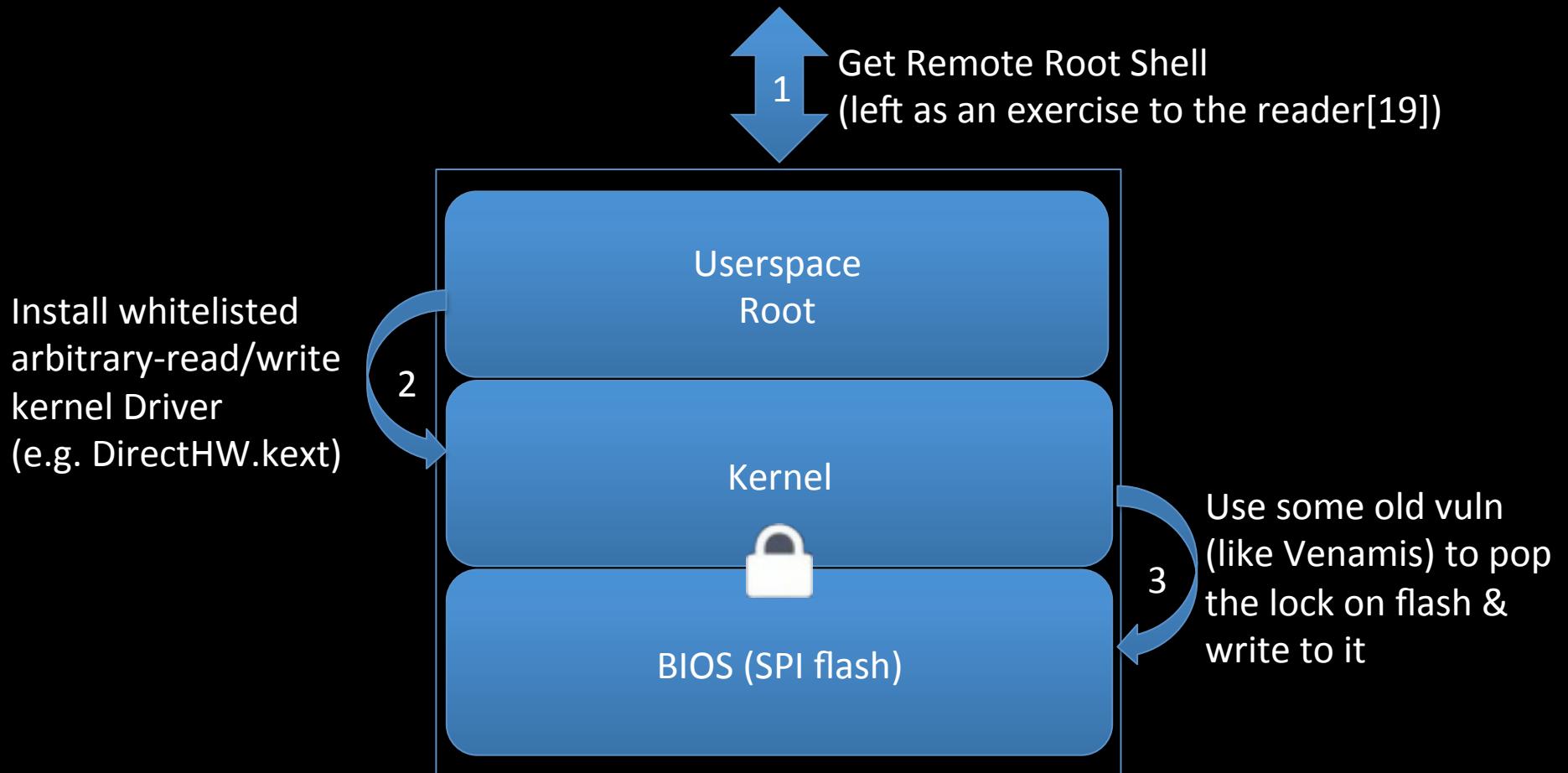
# Attacking Apple firmware

- Remote Apple BIOS infection
- Remote Thunderbolt OROM reflash
- Thunderstrike 2 : Sith Strike
  - A first of it's kind firmworm™ :P

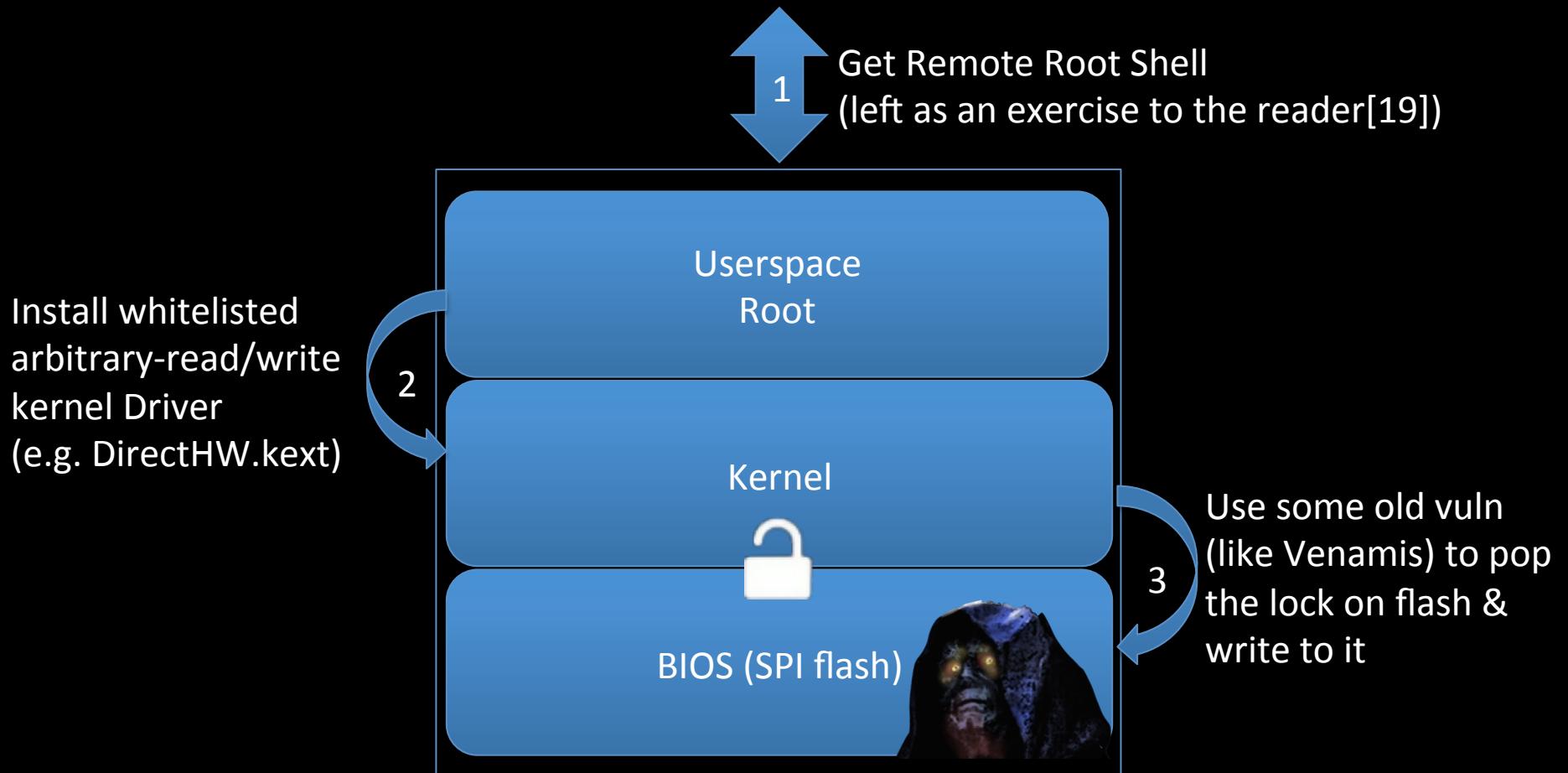
# Remote Apple BIOS infection

- Thunderstrike 1 required physical presence
- We wanted to make it clear *physical presence is not required* to infect Mac firmware

# Remote Apple BIOS infection



# Remote Apple BIOS infection



Rule happily ever after in the Deep Dark,  
knowing **no one ever checks firmware**

# Remote Apple BIOS infection demo

```
mbp2014:~/efi/bh2015: sudo ./check-flockdn
FLOCKDN: f008
PR0:    00000000
PR1:    80010000
PR2:    860f0190
PR3:    9fff0632
mbp2014:~/efi/bh2015: sudo ./install-bootscript unlock-32.bin
000000007ad3f000
00000239 DISPATCH: EntryPoint=000000007af7600
0000029c DISPATCH: EntryPoint=000000007af74fc
Hit ^C to abort

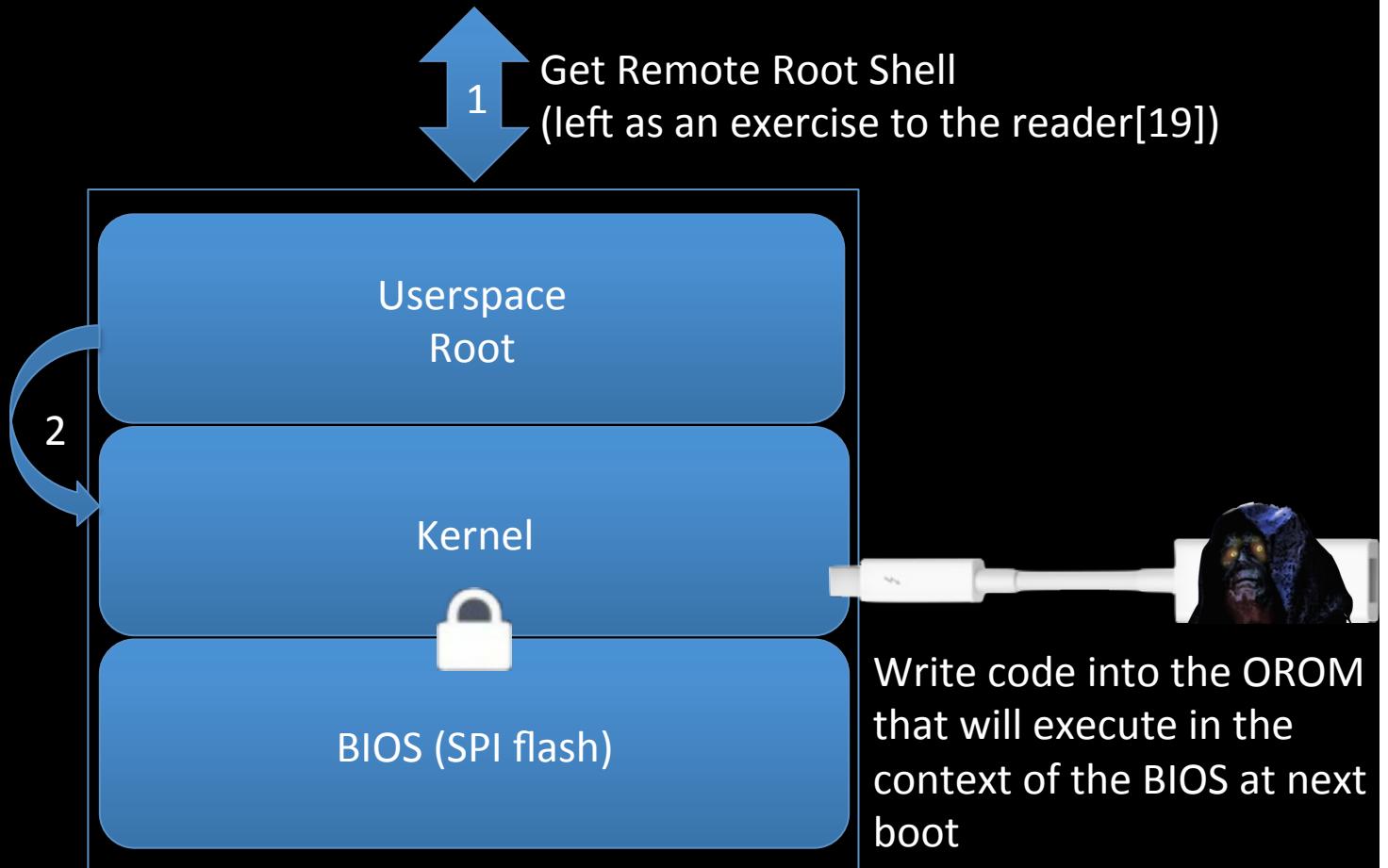
Writing 14 bytes to 0x7af7600
00000000: 66 c7 05 04 f8 d1 fe 08 f0 e9 ee fe ff ff      f.....
mbp2014:~/efi/bh2015: pmset sleepnow
Sleeping now...
mbp2014:~/efi/bh2015: sudo ./check-flockdn
FLOCKDN: f00c
PR0:    00000000
PR1:    00000000
PR2:    00000000
PR3:    00000000
mbp2014:~/efi/bh2015: echo 'Hello, world!' | \
> sudo ./spiflash --verbose -w - --offset 0x7fe000
spiflash_write_enable: bios_ctrl=9
spiflash_write_enable: new_bios_ctrl=9
spiflash: writing to 007fe000: 0xe bytes
spiflash_read: offset 007fe000
spiflash_write: 007fe000 + 1000 bytes
mbp2014:~/efi/bh2015: sudo ./spiflash -r - --offset 0x7fe000 -n 16 | xxd -g 1
00000000: 48 65 6c 6c 6f 2c 20 77 6f 72 6c 64 21 0a ff ff Hello, world!...
mbp2014:~/efi/bh2015: █
```

# Remote OROM reflash

- Snare's talk[20] & Thunderstrike 1 required you to have physical control of the adapter you want to infect, and then boot into FreeDOS to run a Broadcom .exe to update the OROM with a malicious payload
- We wanted to make it clear **physical control of the Ethernet adapter is not required** to infect Thunderbolt to Ethernet OROMs

# Remote Apple OROM infection

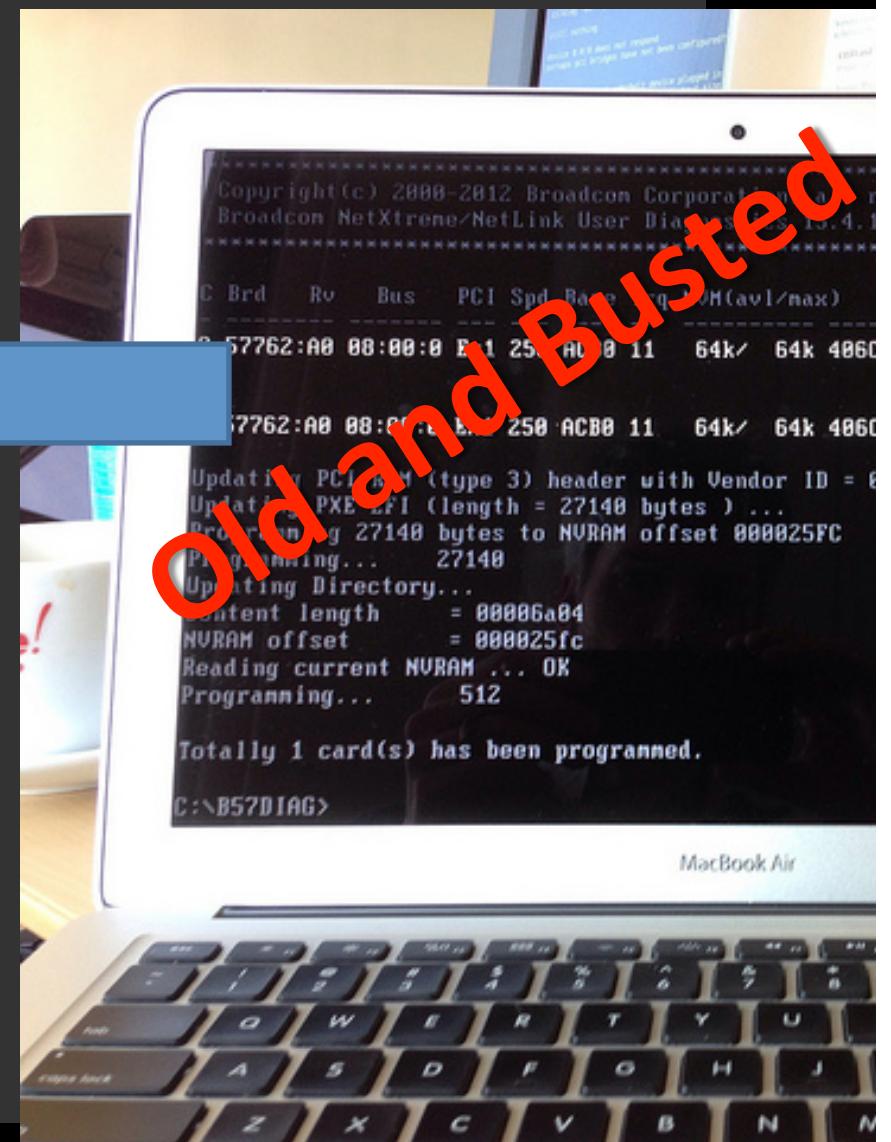
Install a modified version of the Linux “tg3” Broadcom NIC driver



# Remote OROM reflash demo

```
mbp2014:~/efi/bh2015: sudo ./b57tool --pxe hello.rom
Early CRC fc41c8f3 (good)
Header CRC 3c702369 (good)
Header sum dc (good)
MAC: 98:5a:eb:c6:c6:79
Option ROM address 0x25fc length 0x404 bytes
Read 0x400 bytes
PXE CRC e1107f5c
---- new image
Early CRC fc41c8f3 (good)
Header CRC 3c702369 (good)
Header sum dc (good)
MAC: 98:5a:eb:c6:c6:79
Option ROM address 0x25fc length 0x404 bytes
---- writing PXE option rom+crc to 0x25fc
0029fc: 000400 / 000404
---- writing header
0000fc: 0000fc / 000100
---- verify
Early CRC fc41c8f3 (good)
Header CRC 3c702369 (good)
Header sum dc (good)
MAC: 98:5a:eb:c6:c6:79
Option ROM address 0x25fc length 0x404 bytes
mbp2014:~/efi/bh2015:
```

New Hotness



# Thunderstrike 2

- A unique, never before shown “firmworm.”
  - Lives only in firmware
    - Your only option for detection is *firmware forensics*
      - (no one does firmware forensics)
    - Jumps from OROMs to BIOSes and back again

# Thunderstrike 2

Malicious adapter connected



# Thunderstrike 2

System rebooted  
Venamis attack pending



# Thunderstrike 2

System put to sleep  
at some point...



# Thunderstrike 2

System wakes up  
BIOS unlocked. BIOS infected



# Thunderstrike 2

Malicious adapter goes away

...



# Thunderstrike 2

...

Someday a young & naïve  
new adapter arrives



# Thunderstrike 2

BIOS infects all adapters  
It comes in contact with



# Thunderstrike 2

Apprentice leaves  
to infect others



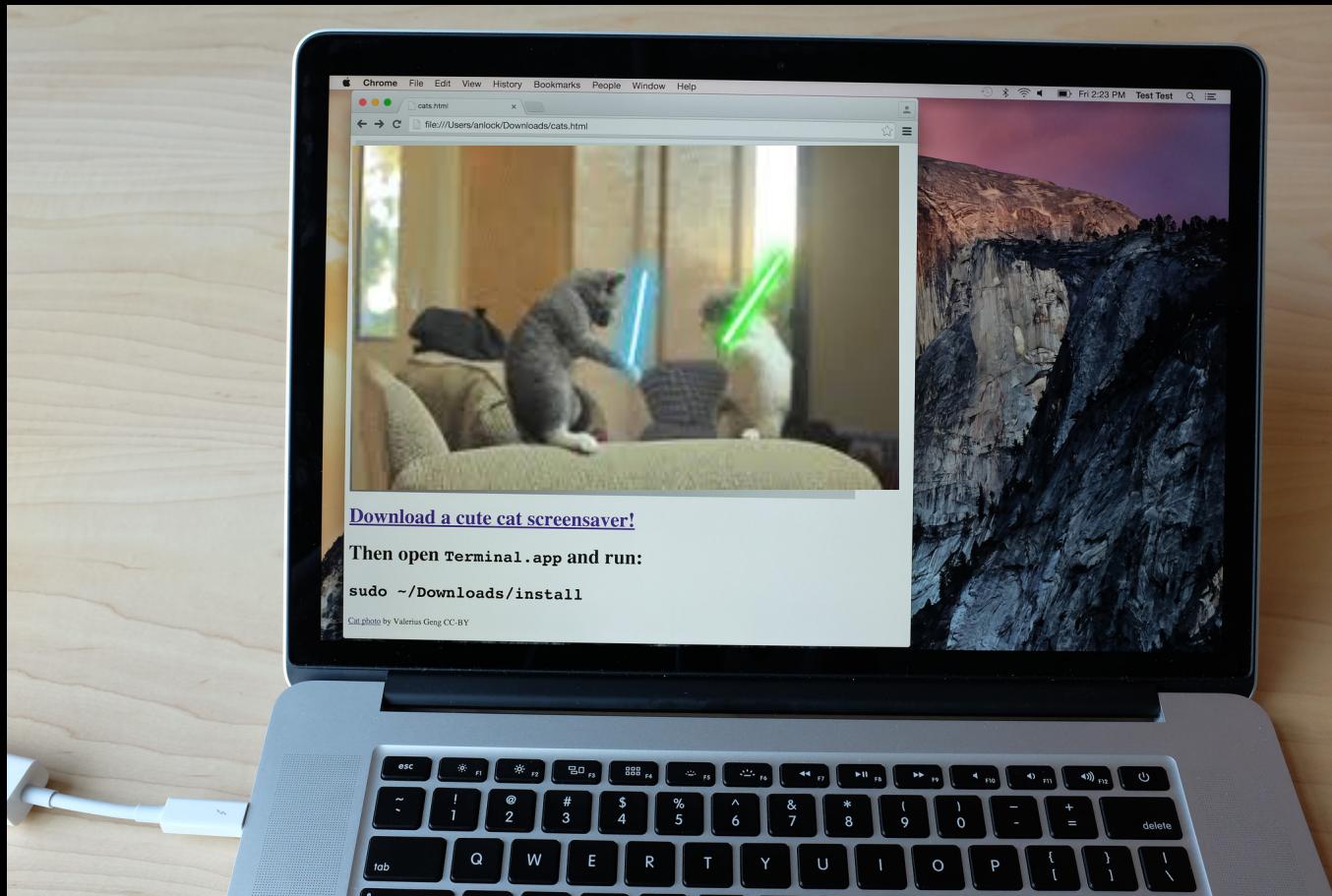
# Thunderstrike 2

Repeat



# Thunderstrike 2

- DEMO VIDEO



# Conclusions



*The dark side of the Force is a pathway to many abilities some consider to be unnatural*

# Reminder: This talk has 1 main point

- Apple has not been as responsive, or as accurate, as other PC vendors in responding to industry-wide notifications of firmware vulnerabilities. Consequently Mac users have been left vulnerable to attacks that have been fixed on other x86-based PCs.

# Time between reporting and patching

- Unknown disclosure:
  - OROM bootkitting: (3 years and counting) Never patched
- Responsible disclosure:
  - Thunderstrike 1: 1.5 years
  - Speed Racer: (1+ year & counting) Never patched
  - The Sicilian: Still investigating exactly which systems it was patched on and when
  - Venamis: partial patch after 9 months (1 month after Prince Harming)
- Full disclosure:
  - Prince Harming: 1 month

# Failing to protect customers

- All firmware vendors have vulnerabilities
- Some vendors are very good about patching public vulnerabilities and auditing for non-public bugs
- Apple is not
- Apple leaves their customers vulnerable to **already-public** issues for long periods of time
- (Until now) Apple got away with reporting they are not vulnerable when they are

# Apple is infectable like everyone else

- Yes, using a Mac is a good way to avoid the *statistical majority* of basic crimeware
  - Yet crimeware still exists for the Mac
- Using a Mac will ***not*** protect you from even moderately sophisticated adversaries
  - Even HackingTeam has a UEFI implant!
    - And now it's "open source" so everyone else does too!
- We have shown numerous ways that Apple firmware can be infected
- *And no one ever checks their firmware*



**KEEP  
CALM  
AND GET YOUR  
SHIT  
TOGETHER**

# Obligatory Band-Aid

- The real fix is to eliminate vulnerabilities at the source. That's what LegbaCore tries to work with vendors to do.
- Failing that...
- A basic OROM dumper will be available from Trammell @ [github.com/osresearch/b57tool](https://github.com/osresearch/b57tool)
- A basic OROM integrity checker is available from Legbacore @ [github.com/legbacore](https://github.com/legbacore)
  - Only tested with our devices, so if you see a difference it could be a false positive to start with. Or it could be real. Send it to us in either case :)

# Contacts & Questions

- Twitter: @qrs, @xenokovah, @coreykal
- Email: hudson@trmm.net, {xeno,corey}@legbacore.com
- <https://trmm.net/PGP> & <http://legbacore.com/Contact.html> for our GPG keys



**OPEN  
SECURITY  
TRAINING  
.INFO**

- Go check out *OpenSecurityTraining.info* for the free classes from Corey and Xeno on x86 assembly & architecture, binary executable formats, stealth malware, and exploits.
- Then go forth and do cool research for us to read about!

One more thing...



# Case study: King & Queen's Gambits

## CERT VU#552286

- “Extreme Privilege Escalation on Windows 8/UEFI Systems” BlackHat USA 2014 Kallenberg et al.
- What’s in a name? It’s also Extreme Privilege Escalation on *Linux* and *Mac OS X* systems!

**Vendor Information** ([Learn More](#))

(Picture retrieved Jun 12<sup>th</sup> 2015)

Vendor	Status	Date Notified	Date Updated
American Megatrends Incorporated (AMI)	Affected	22 Jul 2014	01 Aug 2014
Dell Computer Corporation, Inc.	Affected	22 Jul 2014	28 Oct 2014
Hewlett-Packard Company	Affected	09 Jul 2014	12 Aug 2014
Lenovo	Affected	22 Jul 2014	02 Oct 2014
Phoenix Technologies Ltd.	Affected	22 Jul 2014	28 Oct 2014
Apple Inc.	Not Affected	22 Jul 2014	28 Oct 2014
Insyde Software Corporation	Not Affected	22 Jul 2014	03 Feb 2015
Intel Corporation	Not Affected	03 Dec 2013	19 Sep 2014

# History lesson...

- One VU# for two distinct vulnerabilities
- Found by Corey Kallenberg Nov./Dec. 2013
- Disclosed to Intel/CERT Nov./Dec. 2013
  - Intel took the lead on coordination since they have more vendor contacts than CERT
- Intel patched the EDK2 open source reference code Jan. 2014
  - Intel called/mailed vendors in the Jan.-May timeframe
- Disclosed publicly at BlackHat in Aug. 2014



## LEGBACORE

(Back when we were at MITRE)

<Dec 2013>

Psst...dear firmware makers...exploitable bugs found... Here are the details, go fix it

...



<Aug. 2014>

Dear everyone in the world:  
Vulnerabilities found in UEFI!  
Apply vendor patches!

...

<CanSecWest Mar. 2015>

We want to “work with you” to improve firmware security. Please let us know if we have any vulnerabilities.

Did you fix all the existing vulnerabilities we've disclosed?

We aren't vulnerable to any of that.

Okaaaay....



<Apr. 2015 Confimed they have Venamis vuln>

<May 2015 #PrinceHarming>

<Jun. 2015 #PrinceHarming & Venamis finally patched>

<Jul. 2015>



OK, we did some more tests, and confirmed you're definitely vulnerable to VU#552286

We don't believe you.  
We don't use that code. Prove it.

Sigh, we're really not interested in doing more free work for you. You should have your engineers read the [23 page whitepaper](#) we posted describing the two issues. *Everyone else* who read was able to understand how to patch their systems.

We don't believe you. You haven't given us any proof. You could just be blustering.

I FIND YOUR LACK OF FAITH



DISTURBING



<Jul. 2015>



**SIGH!**

- \* Here is a PoC
- \* Set a breakpoint at 0xFFFF15245 in your hardware debugger.
- \* The second time the breakpoint fires, look at the arguments to this memcpy function.
- \* Observe *the arbitrary size memcpy with arbitrary attacker-controlled contents* that is about to take place.

OK. We'll look into it.

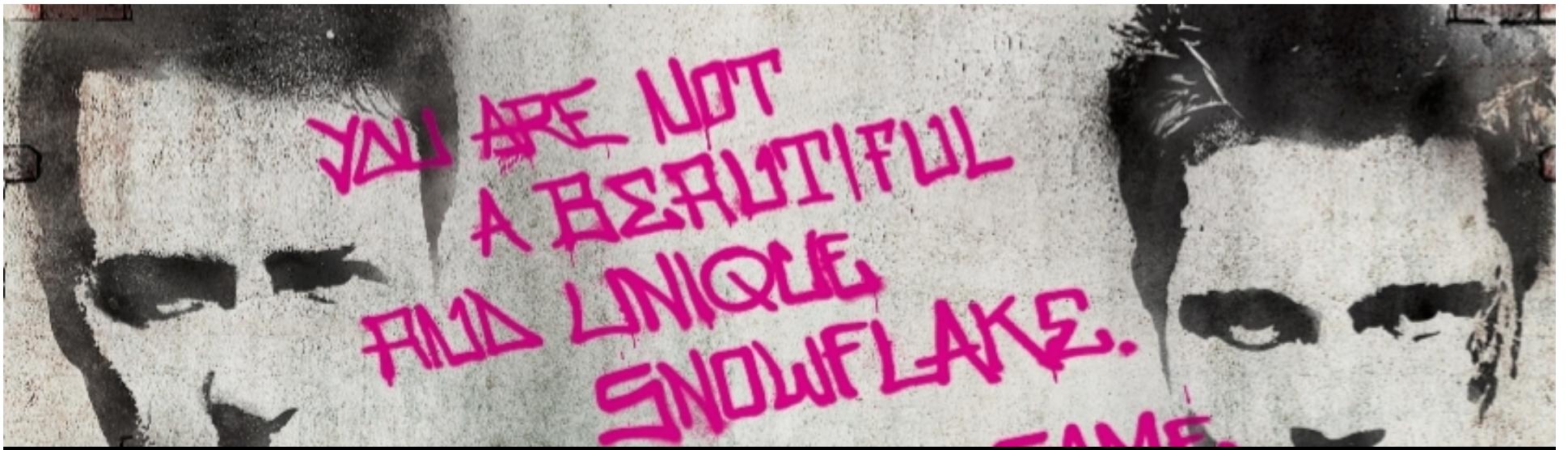
# And *THAT'S* the story...

- Of how our *two-person startup* spent part of our summer doing *free work* for a company with *multibillion dollar quarterly profits*, because they *couldn't/wouldn't* fix a *responsibly disclosed* firmware vulnerability that's been *public for a year now*, that we *confirmed* existed *on my wife's MacBook Air!*
- Good times!
- A security win!
- High fives all around!
- Let's hear it for the (90s) status quo!

An open letter to firmware makers:

ALL (U)EFI-RELATED FIRMWARE  
LOOKS THE SAME TO ATTACKERS!

YOUR SECURITY THROUGH  
OBSCURITY IS GONE!



MacBookAir4,1

Function name

```

f sub_79E2124
f sub_79E2126
f sub_79E212B
f sub_79E212E
f sub_79E2130
f sub_79E2132
f sub_79E2132
f sub_79E2133
f sub_79E2134
f sub_79E2135
f sub_79E2136
f sub_79E2136
f sub_79E2137
f sub_79E2138
f CopyMem
f sub_79E2162
sub_79E2165
f SetMem
f sub_79E2171
f sub_79E2171

```

Line 50 of 245

Gr... □ ×

loc\_79E2B672:
mov rcx, [r15+0A0h]
call CoreRestoreTp1
mov rcx, [r15+0C0h]
call CoreFreePool
mov cs:qword \_79E315D8, r12
mov rcx, r14
call sub\_79E28F12
test rsi, rsi
jz short loc\_79E2B6B9

test rdi, rdi
jz short loc\_79E2B6B9

mov r14, [r15+0B0h]
mov [rdi], r14
mov r14, [r15+0B0h]
mov [rsi], r14
jmp short loc\_79E2B6D0

loc\_79E2B6B9:
mov rcx, [r15+0B0h]
call CoreFreePool
mov qword ptr [r15+0B0h], 0

loc\_79E2B6D0:
mov r14, [r15+0A0h]
test r14, r14
js short loc\_79E2B6E3

100.00% (215,1648) (118,170) 0000A679 000000079E2B679: CoreStartImage+159

AU: idle Down Disk: 15GB

ASUS BT1AH (from our CanSecWest 2015 talk)

Function name

```

f sub_1800002C0
f sub_18000030C
f sub_180000358
f sub_180000414
f sub_18000041C
f sub_180000428
f sub_180000460
f sub_18000049C
f sub_180000590
f sub_180000620
f sub_180000704
f sub_180000758
f sub_1800007E8
f sub_180000848
f sub_1800008A8
f sub_1800008E8
f sub_180000900
f sub_18000092C
f sub_180000A40

```

Line 553 of 1209

Graph ov... □ ×

loc\_18001EBD7: ; supposed to be Image->Tp1
mov rcx, [rbx+LOADED\_IMAGE\_PRIVATE\_DATA.Tp1]
call CoreRestoreTp1
mov rcx, [rbx+LOADED\_IMAGE\_PRIVATE\_DATA.JumpContext] ; supposed to
call CoreFreePool
mov rcx, r12 ; r12 = HandleDatabaseKey
mov cs:mCurrentImage, r13
call CoreConnectHandlesByKey
test rsi, rsi ; rsi = ExitData
jz short loc\_18001EC1E ; supposed to be Image->ExitData

test rdi, rdi ; rdi = ExitDataSize
jz short loc\_18001EC1E ; supposed to be Image->ExitData

osed to be Image->ExitDataSize
osed to be Image->ExitData

Status = Image->Status

loc\_18001EC1E: ; supposed to be
mov rcx, [rbx+0C0h]
call CoreFreePool
and qword ptr [rbx+0C0h], 0 ; suppose

; Status = Image->Status
0000000000000000
18001EC4F

100.00% (634,1707) (2,107) 0001EBDE 000000018001EBDE: CoreStartImage+13A

AU: idle Down Disk: 15GB

# Failing to protect its customers

- *All* firmware vendors have vulnerabilities
- Some vendors are *very good* about patching public vulnerabilities and auditing for non-public bugs
- *Apple is not*
- Apple leaves their customers vulnerable to **already-public** issues for long periods of time
- (Until now) Apple got away with reporting they are not vulnerable when they are

# Thanks for attending our talk!

- We need to make way for the next speakers
- Please meet us outside the room for Q&A



# References

- [0] Attacking Intel BIOS – Rafal Wojtczuk and Alexander Tereshkin, July 2009  
<http://invisiblethingslab.com/resources/bh09usa/Attacking%20Intel%20BIOS.pdf>  
<http://www.kb.cert.org/vuls/id/127284> (CERT never posted?!)
- [1] Defeating Signed BIOS Enforcement – Kallenberg et al., Sept. 2013  
<http://conference.hitb.org/hitbsecconf2013kul/materials/D1T1%20-%20Kallenberg,%20Kovah,%20Butterworth%20-%20Defeating%20Signed%20BIOS%20Enforcement.pdf>  
<http://www.kb.cert.org/vuls/id/912156>  
<http://www.kb.cert.org/vuls/id/255726> (CERT hasn't posted yet despite request)
- [2] All Your Boot Are Belong To Us (MITRE portion) – Kallenberg et al. – Mar. 2014, delayed from publicly disclosing potential for bricking until HITB at Intel's request  
[https://cansecwest.com/slides/2014/AllYourBoot\\_csw14-mitre-final.pdf](https://cansecwest.com/slides/2014/AllYourBoot_csw14-mitre-final.pdf)  
<http://www.kb.cert.org/vuls/id/758382>
- [3] All Your Boot Are Belong To Us (Intel portion) – Bulygin et al. – Mar. 2014  
[https://cansecwest.com/slides/2014/AllYourBoot\\_csw14-intel-final.pdf](https://cansecwest.com/slides/2014/AllYourBoot_csw14-intel-final.pdf)
- [4] Setup for Failure: Defeating UEFI Secure Boot - Kallenberg et al., Apr. 2014  
[http://syscan.org/index.php/download/get/6e597f6067493dd581eed737146f3afb/SyScan2014\\_CoreyKallenberg\\_SetupforFailureDefeatingSecureBoot.zip](http://syscan.org/index.php/download/get/6e597f6067493dd581eed737146f3afb/SyScan2014_CoreyKallenberg_SetupforFailureDefeatingSecureBoot.zip)  
<http://www.kb.cert.org/vuls/id/291102> (CERT hasn't posted yet despite request)

# References

- [5] Extreme Privilege Escalation on UEFI Windows 8 Systems – Kallenberg et al., Aug. 2014  
<https://www.blackhat.com/docs/us-14/materials/us-14-Kallenberg-Extreme-Privilege-Escalation-On-Windows8-UEFI-Systems.pdf>  
<http://www.kb.cert.org/vuls/id/766164>
- [6] Attacks against UEFI Inspired by Darth Venamis and Speed Racer – Wojtczuk & Kallenberg, Dec. 2013  
[https://bromiumlabs.files.wordpress.com/2015/01/attacksonuefi\\_slides.pdf](https://bromiumlabs.files.wordpress.com/2015/01/attacksonuefi_slides.pdf)  
<http://www.kb.cert.org/vuls/id/533140>
- [7] Speed Racer: Exploiting an Intel Flash Protection Race Condition – Kallenberg & Wojtczuk, Dec. 2013  
[https://frab.cccv.de/system/attachments/2565/original/speed\\_racer\\_whitepaper.pdf](https://frab.cccv.de/system/attachments/2565/original/speed_racer_whitepaper.pdf)  
<http://www.kb.cert.org/vuls/id/912156>
- [8] Attacking UEFI Boot Script – Wojtczuk & Kallenberg, Dec. 2013  
[https://frab.cccv.de/system/attachments/2566/original/venamis\\_whitepaper.pdf](https://frab.cccv.de/system/attachments/2566/original/venamis_whitepaper.pdf)  
<http://www.kb.cert.org/vuls/id/552286>
- [9] “Snorlax” bug – Cornwell, et al., Dec. 2013  
[https://frab.cccv.de/system/attachments/2566/original/venamis\\_whitepaper.pdf](https://frab.cccv.de/system/attachments/2566/original/venamis_whitepaper.pdf)  
<http://www.kb.cert.org/vuls/id/577140>

# References

[13] ThunderStrike, Trammell Hudson,  
<https://trmm.net/Thunderstrike>

[14] “The Empire Strikes Back Apple – how your Mac firmware security is completely broken”, Pedro Viliaca,  
<https://reverse.put.as/2015/05/29/the-empire-strikes-back-apple-how-your-mac-firmware-security-is-completely-broken/>

[17] “Exploiting UEFI boot script table vulnerability”, Dmytro Oleksiuk,  
<http://blog.cr4.sh/2015/02/exploiting-uefi-boot-script-table.html>

[18] Boot script vulnerability proof of concept code, Dmytro Oleksiuk,  
[https://github.com/Cr4sh/UEFI\\_boot\\_script\\_expl](https://github.com/Cr4sh/UEFI_boot_script_expl)

[19] OS X Yosemite v10.10.4 and Security Update 2015-005 (69 CVEs)  
<https://support.apple.com/en-us/HT204942>

[20] DE MYSTERICIIS DOM JOBSIVS Mac EFI Rootkits - Loukas K (snare),  
Jul. 2012  
[https://media.blackhat.com/bh-us-12/Briefings/Loukas\\_K\\_BH\\_US\\_12\\_LoukasK\\_De\\_Mystericis\\_Dom\\_Jobsivs\\_Slides.pdf](https://media.blackhat.com/bh-us-12/Briefings/Loukas_K_BH_US_12_LoukasK_De_Mystericis_Dom_Jobsivs_Slides.pdf)

# References

- [21] CIH (Computer Virus), Wikipedia,  
[https://en.wikipedia.org/wiki/CIH\\_%28computer\\_virus%29](https://en.wikipedia.org/wiki/CIH_%28computer_virus%29)
- [22] Mebromi: the first BIOS rootkit in the wild, Marco Giuliani,  
<http://www.webroot.com/blog/2011/09/13/mebromi-the-first-bios-rootkit-in-the-wild/>
- [23] Shopping for Spy Gear: Catalog Advertises NSA Toolbox, Appelbaum et al.,  
<http://www.spiegel.de/international/world/catalog-reveals-nsa-has-back-doors-for-numerous-devices-a-940994.html>
- [24] Nikolaj Schlej's twitter post about HackingTeam UEFI malware,  
<https://twitter.com/NikolajSchlej/status/618076694117789696>
- [25] Analyzing Malware for Embedded Devices: TheMoon Worm, Bernardo Rodrigues,  
<http://w00tsec.blogspot.com/2014/02/analyzing-malware-for-embedded-devices.html>

# References

[X] See all the related work we're aware of here:

[http://timeglider.com/timeline/  
5ca2daa6078caaf4](http://timeglider.com/timeline/5ca2daa6078caaf4)

# Backup

- “Should you worry when the skullhead is in front of you? Or is it worse because it’s always waiting, where your eyes don’t go?”  
– They Might Be Giants



# FAQ

- What is your talk about?
  - How Apple has not been as responsive, or as accurate, as other PC vendors in responding to industry-wide notifications of firmware vulnerabilities. How, consequently Macs have been left vulnerable to attacks that have been fixed on other x86-based PCs. And how malware can infect Mac firmware without physical presence just as easily as we've shown in the past on PCs, or via a novel 'firmworm' propagation utilizing Apple's Thunderbolt-to-Ethernet adapter.
- How were vulnerabilities disclosed to Apple?
  - They were disclosed to *all PC vendors* in parallel through coordinated disclosure efforts led by Intel, CERT-CC, and the UEFI Security Response Team (once it was established in 2014) in 2013-2014.
    - As shown on the respective CERT VU# pages, Apple either said it was not vulnerable to any of them, or Apple did not respond at all
  - When this did not seem to be working, we started emailing product-security@apple.com directly.

# FAQ

- Do attackers require physical access to infect the PCI Option ROM (OROM) or BIOS?
  - No
- What models are affected?
  - All Apple Thunderbolt to Ethernet adapters' OROMs that we have tested can be rewritten to include malicious code. And architecturally all Apple BIOSes will execute the attacker's code from the OROM in the context of the BIOS.
  - There are other OROMs in the system, although our proof of concept only writes to the Ethernet adapter.
  - You would have to see advisories from Apple to determine what is affected by the different BIOS-specific vulnerabilities that allow BIOS infection. At least for the Venamis vulnerability (CERT VU#). Basically all Macs since 2011 were patched.
- Are PCs vulnerable as well?
  - Yes. All of the vulnerabilities described in this talk were first found on other PCs, in some cases more than a year ago. They're just all vulnerabilities which Apple never patched. Many low-end PC makers have also failed to patch the issues as well.

# FAQ

- How easy is it for attackers to exploit these vulnerabilities?
  - We would say a medium level of difficulty. It isn't trivial, and requires the attackers to have some foundational understanding of how computers work (which script kiddies for instance will generally lack. And we never release exploit code that would enable script kiddies.) But at the same time, these attacks do not require jumping through anywhere near as many hoops as modern browser/OS exploits for instance. There are no exploit mitigations within BIOSes.

# FAQ

- What are the consequences of a firmware attack?
  - An attacker can execute arbitrary code on the system at the earliest stages. An attacker that runs first on the system is the most powerful and privileged attacker.
  - They can persist past boot and manipulate the OS during runtime, by modifying the OS's memory before it starts, or by living in System Management RAM (SMRAM), which architecturally cannot be inspected by security software.
  - An attacker can also render the system un-bootable (“bricked”) in a way that it cannot be recovered through any normal HD/SSD replacement, or reinstalling via external USB/DVD
  - By persisting in the firmware, rather than in the filesystem of an OS, an attacker can maintain stealth control of a system longer, because prior to LegbaCore’s founding, there were no computer security companies that focused on inspecting PC firmware for attackers.
  - More concrete examples of what an attacker can do are shown in our past work at [legbacore.com/Research.html](http://legbacore.com/Research.html)

# FAQ

- What do you mean when you say “Thunderstrike 2 is the ‘first of its kind’ ‘firmworm’”?
  - There have been other worms that have spread from one device’s firmware to another’s – specifically in the area of SOHO routers [25]. However the ‘firmware’ that is being used in those cases is just an embedded version of Linux, and therefore in practice the malware does not look any different from past worms that jumped from OS to OS. And defenders already have a plethora of tools to inspect OSes like Linux for malware.
  - Stuxnet was an OS-to-OS worm that had a SCADA firmware infecting component. But it could not propagate without residing within the Windows OS as a kernel driver, or on the filesystem of USB drives as an exploit. This is why it was ultimately detected.
  - Thunderstrike 2 is unique in that it can propagate without *\*ever\** leaving any evidence in the areas which are typically inspected: OS memory and OS filesystem. It will live in the PCI Option ROM firmware on an Apple thunderbolt-to-ethernet adapter, and it will live in the EFI firmware flash chip that runs before the OS. As such, security products and forensic examiners are woefully unprepared to detect/disinfect this class of malware, and would never be able to find or remove it, if we weren’t making tools to do so.
  - We are **not** saying such attacks are undetectable. Far from it, they are definitely detectable. But no one is *trying* to detect them, and consequently sophisticated attackers know that if they dive this deep in the system, it will have maximum privilege to do whatever they want, and have the freedom to operate undetected.

# FAQ

- Are you aware of people attacking firmware in the wild?
  - The first malware detected in the wild that *wiped* the BIOS (to brick a system) was CIH[21] in 1998
  - The first malware detected in the wild that used the BIOS to re-infect wiped/replaced hard drives was Mebromi[22] in 2011.
  - In 2013 leaks by Edward Snowden indicated that NSA has had BIOS/HD firmware infection capability since at least 2008[23]
  - When the surveillance software vendor HackingTeam was hacked in 2015, it was discovered in their leaked files[24] that they had been selling a capability to infect UEFI firmware and drop their existing Windows backdoor.
    - The source code for this UEFI rootkit was also leaked, and therefore it is assumed that all other surveillance software companies that didn't already have such a capability, will now be able to pursue one with less effort.
  - The reader is also recommended to see the magnitude of conference talks that have been given on the topic[X]. Some of this research, like eEye BootRoot was ultimately weaponized and detected in the wild, and others, like BluePill, are highly suspected to be in the wild as targeted attacks[NSA], but have never been \*actually\* detected in the wild (due to the failings of security software.)

# FAQ

- Are you aware of in the wild attacks targeting Apple's firmware specifically?
  - No. Because we are not aware of any companies collecting the data necessary to detect firmware attackers on Mac. And tools to integrity check firmware did not exist until we created them relatively recently.
- Why doesn't Apple follow Intel's recommendations to firmware makers and use all the protection mechanisms available to them?
  - We don't know for sure, although their firmware is based on the much older 1.10 EFI release instead of the current 2.x UEFI tree.
  - We speculate it is because they design their architecture a bit differently from other vendors, putting more reliance on their "System Management Controller" (SMC) chip, and running less code in the x86 System Management Mode (SMM). (We have not specifically investigated the relative amount of code Apple runs in SMM vs. other PC vendors.) Therefore they think that SMM compromises are less damaging. However SMC compromises have been shown by other researchers.[ionescue]

# FAQ

- What can Apple customers do to protect themselves from firmware attacks?
  - Hope that Apple releases patches, and then apply the patches through the normal software update mechanism
- What can Apple customers do to determine if their firmware is already infected?
  - For the OROM infection shown in the talk, use the cited OROM dump & integrity check tools
  - For the main BIOS infection itself, *trustworthy* integrity checking (which provides truth in the face of the lies that attackers can feed to security software) is more complicated and labor-intensive. A such, LegbaCore performs such checks as a paid service.

# FAQ

- Is ThunderStrike 2 fixed by OS X Yosemite v10.10.4 and Security Update 2015-005?
  - Apple released a partial patch, that fixed the main Venamis and PrinceHarming vulnerabilities by locking the Protected Range Registers (PRRs) and FLOCKDN in PEI before the S3 boot script is executed.
    - A hooked S3 sleep resume script can no longer directly configure the flash to be writable.
    - The attacker can still use a hooked boot script to disable SMRRs, and therefore break into System Management Mode.
  - This patch is insufficient and can be bypassed by sophisticated attackers to re-disable PRRs, and re-enable writing to the firmware.

# FAQ

- Does Secure Boot prevent Thunderstrike 2?
  - Apple does not implement UEFI SecureBoot as outlined in the UEFI 2.3.1 specification.
  - Even if they did, the types of existing public vulnerabilities described in this talk are the types which can circumvent SecureBoot.
  - Implementation of SecureBoot *would* add a requirement for signatures on Option ROMs, which would prevent a random Thunderbolt device from being able to inject untrusted code into the boot process.

# FAQ

- Does a firmware password prevent this attack?
  - No
- Where can I get the PoC attack binary/source code?
  - It is not being made available
- Is this related to Inception?
  - Inception uses DMA to read/write memory, Thunderstrike is about writing to the boot flash.

# FAQ

- What are Option ROMs?
  - The original IBM PC included Option ROM sockets for optional features. These were typically used for things like "ROM BASIC" or drivers for additional devices. At boot time, the BIOS would see if there were any chips installed and execute startup code from them. Adapter cards in the ISA bus could also provide expansion ROMs with code to initialize themselves. This legacy feature was carried through all the way to the modern PCI Express (PCIe) bus -- when a modern computer boots it queries every device installed to check if there is code to be executed before loading the operating system. Thunderbolt brings the PCIe bus to the outside and allows external devices to provide Option ROMs to the EFI firmware.

# FAQ

- {Computers used to have/My computer has} a physical hardware switch that stopped the BIOS from being writeable, unless the user physically modified the switch. Why don't computer makers still do that?
  - You'd have to ask the vendors. We can only speculate.
  - One reason could be because firmware has gotten so large and complex, that firmware updates are inevitable, and vendors want to make it as easy for their customers to apply them as possible.
  - Another reason could be that it is difficult to engineer-in a writability switch which is deemed aesthetically acceptable. And the associated user education that goes with the proper usage of the switch would cause more customer support calls (at more cost), and would very likely
    - Though we would note that Google ChromeBooks take an elegant approach of using a screw to render the firmware write-protected. And then if the screw is removed (voiding the warranty), the firmware is then writable if the users would like to customize the system.
    - However this would likely not be a viable approach for Apple's hardware, as Apple discourages its customers from being able to open their hardware.
  - Another reason is that the removal of even a single component that is deemed to not have high customer demand (i.e. a "BIOS write enable" switch) can lead to significant cost savings for PC vendors, and PCs are a very low-margin business.
  - Ultimately, even if vendors added switches, people would still be vulnerable to malware masquerading as a legitimate software and social engineering the user into marking their firmware as writable, before infecting it (with or without an exploit.)
- We want to convey to those who would suggest it, that the solution is \*not\* "just add a switch." That only provides a false sense of security. The actual fix for firmware security is a more subtle mix of rearchitecting the firmware with some combination of true ROM reset vectors, white-box trusted hardware, cryptographically verified signatures on updates, measured boot, and other related technologies. As such, we anticipate that the final goal will not be reached for many years. However that is why LegbaCore is working with vendors to lend our knowledge of attacks to get to the end state, of *an actually-trustworthy foundation*, faster.