MIRcon 2014

Into the Unknown: Assessing your BIOS Vulnerabilities

Xeno Kovah
Lead InfoSec Engineer
MITRE

Research Team – Credit Where Credit's Due


Corey Kallenberg


John Butterworth

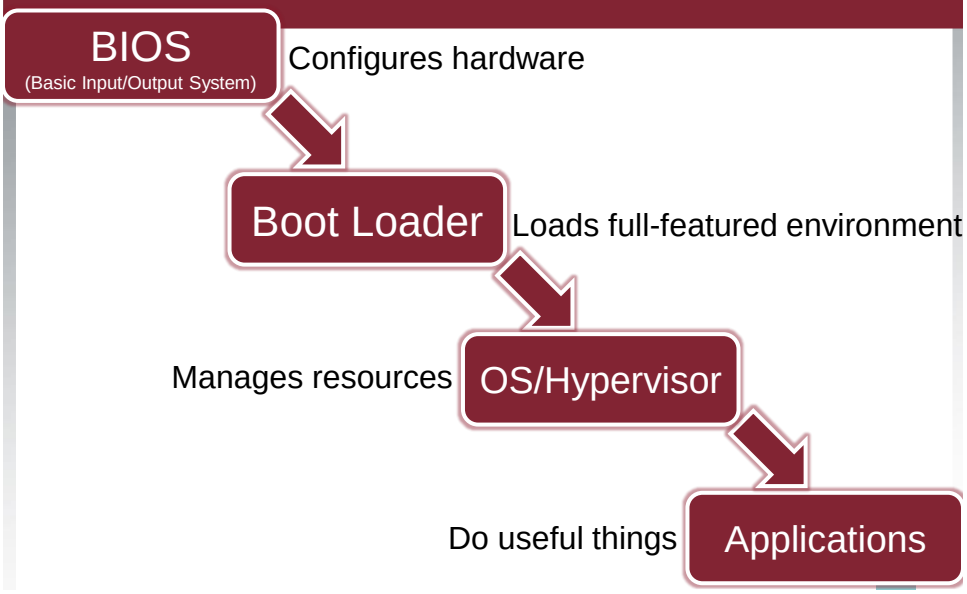

Sam Cornwell


Bob Heinemann

Introduction

- Who we are:
  - Trusted Computing and firmware security researchers at The MITRE Corporation
- What MITRE is:
  - A not-for-profit company that runs seven US Government "Federally Funded Research & Development Centers" (FFRDCs) dedicated to working in the public interest
  - Technical lead for a number of standards and structured data exchange formats such as *CVE*, CWE, OVAL, CAPEC, STIX, TAXII, etc
  - The first .org, !(.mil | .gov | .com | .edu | .net), on the ARPANET

How computers do useful things



One of these things is not like the others

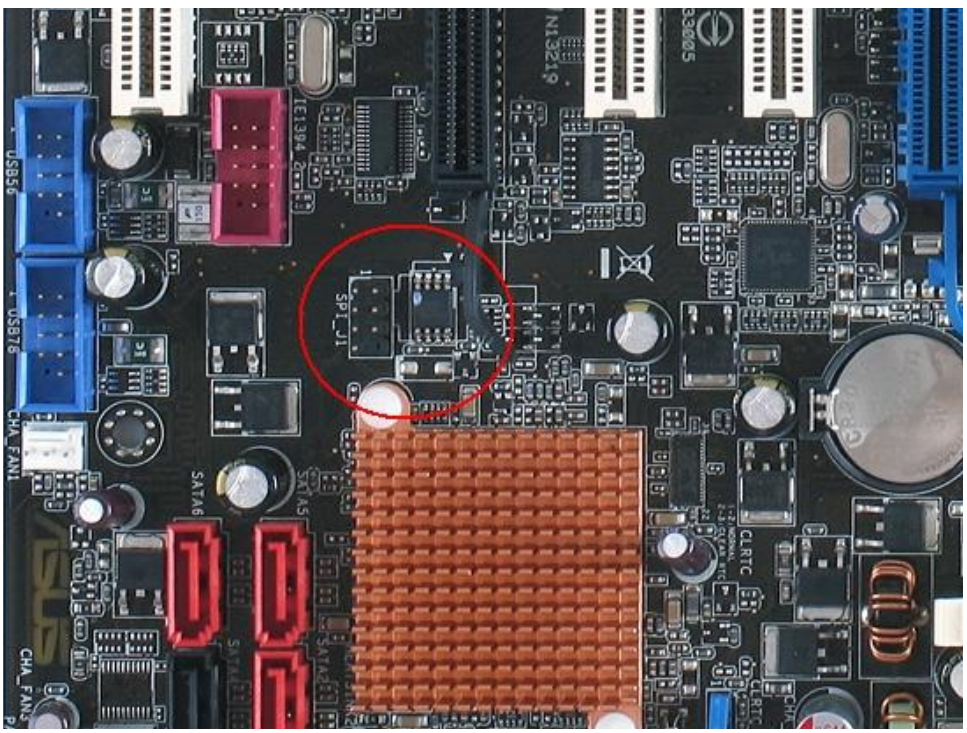
One of these things is not well understood

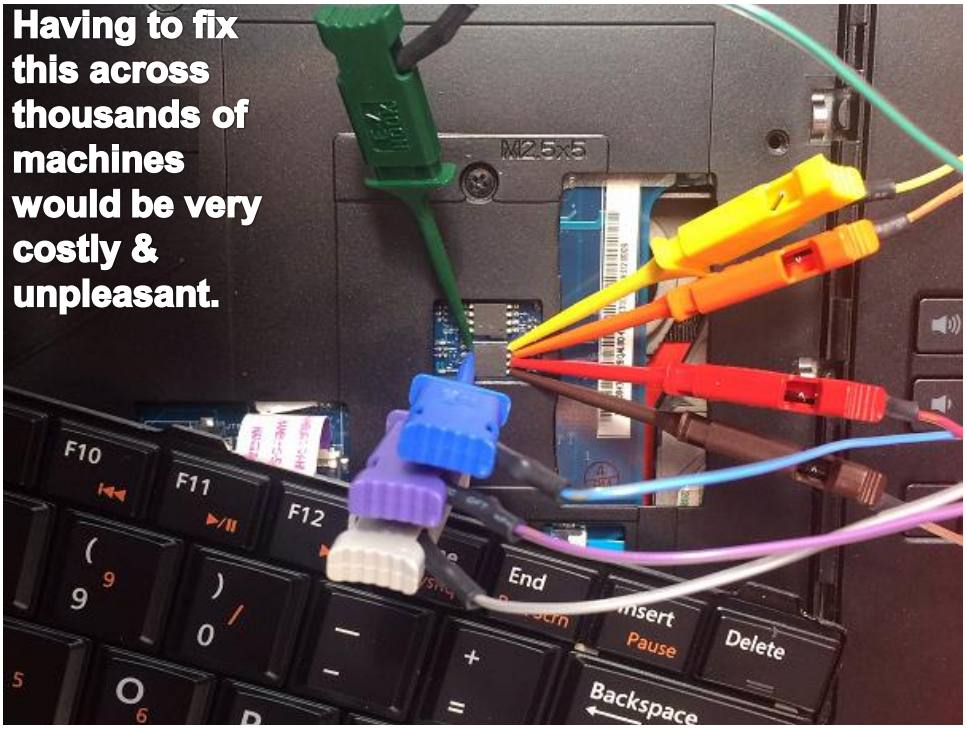
BIOS (Basic Input/Output System)
Firmware
Lives in flash chip soldered to motherboard

Boot Loader
Software
Lives on hard drive

Software
Lives on hard drive
OS/Hypervisor

Software
Lives on hard drive
Applications/OS





What you don't know *can* hurt you

- BIOS is the first code that the CPU ever runs.
  - It can affect and compromise all subsequent code that runs
  - It is a black box that almost no one understands
- Therefore we needed to become BIOS SMEs and share our knowledge and findings with others

Highlights of what we have found in our short time working on firmware security

- There were no public tools to confirm BIOS access controls were set properly
  - And public tools to even *read* the BIOS were spotty at best!
  - So we made one, "Copernicus", and made the binary freely available so anyone could check their system [26]
- A key Trusted Computing Group technology that supported a secure boot up (the Static Core Root of Trust for Measurement) was weak[18]. But we could strengthen it with our previous work [19]
- We found, disclosed, and saw patched the second ever publicly talked about BIOS exploit [13]
  - Patched by Dell 7/2013, affected 22 Legacy BIOS models CVE-2013-3582, CERT VU# 912156

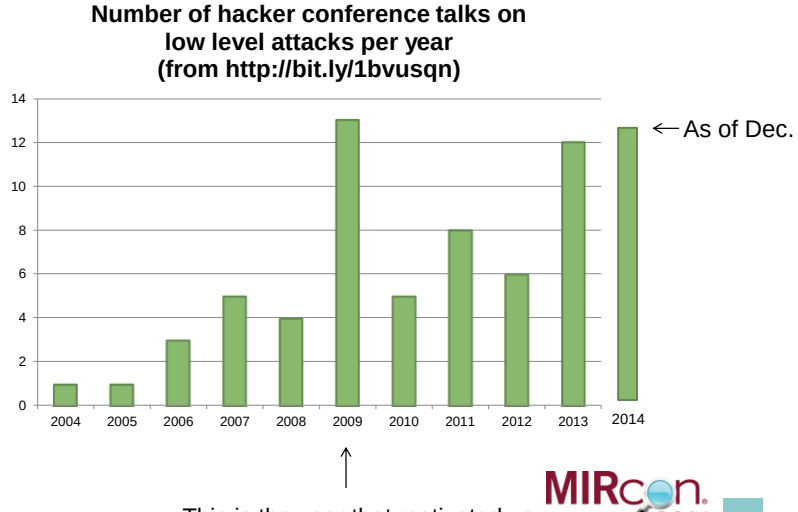
Highlights of what we have found in our short time working on firmware security

- Discovered a new type of Man in the Middle (MitM) attack that could universally hide from software-based BIOS integrity checkers
  - "SMM MitM" attacker dubbed "Smite'em the Stealthy" [27]
  - We made "Copernicus 2" using Intel Trusted Execution Technology to combat Smite'em [28]
- Problems with Unified Extensible Firmware Interface (UEFI) variables that could lead to bypassing Windows 8 SecureBoot [29]

Highlights of what we have found in our short time working on firmware security

- 2 confirmed-exploitable buffer overflows in the open source Intel reference UEFI BIOS implementation [31]
  - CVE-2014-4859 & CVE-2014-4860, CERT VU # 552286
  - This reference code is used by many other OEMs
  - Affects Intel, Phoenix, AMI, HP (affected > 500 models), Dell (some of 39 affected models are patched), Lenovo (TBD models)
  - Insyde say they're not vulnerable.
  - Waiting for patches from other vendors.
- And more things still under disclosure moratorium

Knowledge about low level attacks is growing



One Stealth Malware Taxonomy

aka "Why would someone bother with a firmware attack?" (answer: maximum power)

- Ring 3 – Userspace-Based
- Ring 0 – Kernel-Based
- "Ring -1" – Virtualization-Based
  - Intel VT-x(Virtualization Technology for x86), AMD-V (AMD Virtualization), Hypervisor subverted
- "Ring -1.5?" - Post-BIOS, Pre OS/VMM
  - e.g. Master Boot Record (MBR) "bootkit"
  - Peripherals with DMA(Direct Memory Access) (this can be ring 0, -1, or -1.5 depending on whether VT-d is being used)
  - Not a generally acknowledged "ring", but the place I think it fits best
- "Ring -2" – System Management Mode (SMM)
- "Ring -2.25 – SMM/SMI Transfer Monitor (STM)
  - A hypervisor dedicated to virtualizing SMM
  - Another one of my made up "rings", I just added this ring for this presentation :)
- "Ring -2.5" - BIOS (Basic Input Output System), EFI (Extensible Firmware Interface)
  - because they are the first code to execute *on the CPU* and they control what gets loaded into SMM
  - Not a generally acknowledged "ring", but the place I think it fits best
- "Ring -3" – Chipset Based - *probably not valid anymore on modern architectures*
  - Intel AMT(Active Management Technology)
  - Could maybe be argued that any off-CPU, DMA-capable peripherals live at this level?

So What?

- What are some consequences of firmware attacks?

Example Attacks
BIOS "Bricking"

Firmware is corrupted (1 byte is all that's needed)
System will not boot

The CIH virus did this as a time-bomb attack on (*supposedly* 60 million) computers in 1998http://en.wikipedia.org/wiki/CIH_(computer_virus)

Example Attacks
BIOS Backdooring

This attack was seen in China in 2011 in the Mebromi malware.
http://www.webroot.com/blog/2011/09/13/mebromi-the-first-bios-rootkit-in-the-wild/

Example Attacks
Uber Evil Maid

BIOS
(Basic Input/Output System)

Hook Full Disk Encryption (FDE) software in memory

Boot Loader &
FDE Prompt

Capture FDE password
Store in BIOS or unused HD space

Steal full HD contents at one's leisure

Example Attacks
Cloud Bursting

BIOS
(Basic Input/Output System)

Infect Boot Loader

Boot Loader

Infect hypervisor binary or infect virtual BIOS

Infect ALL virtual machines

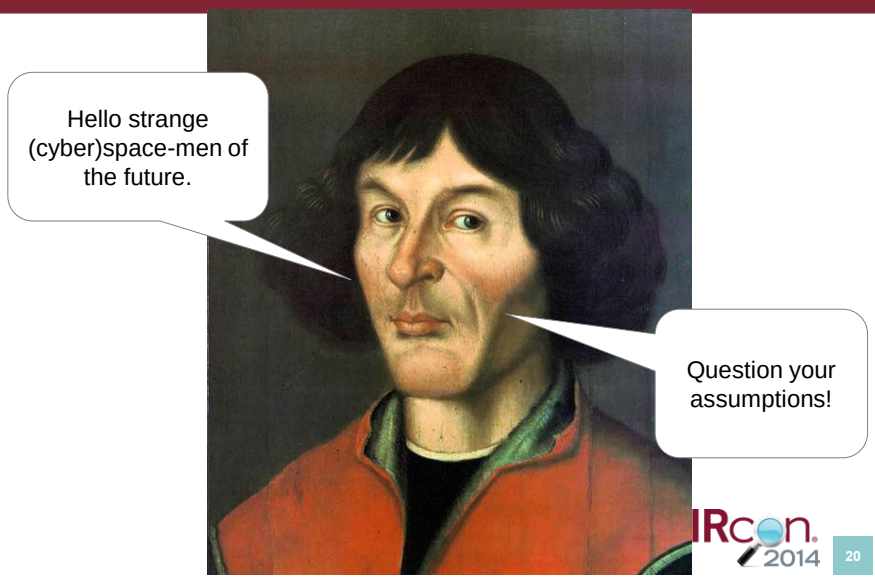
Hypervisor

Steal or smash the eggs

Existing "Best-Effort" PC Firmware Checking Capabilities

- MITRE Copernicus
  - Targeted at enterprise deployment, run on all of MITRE's systems
- Intel Chipsec
  - Targeted at researcher experimentation & OEM's checking systems before shipping them
- Intel OpenAttestation
  - Attests to measurements stored in TPM Platform Configuration Registers
- Flashrom
  - For firmware read/write from as many platforms as possible
    - Doesn't support most modern hardware.
- Built on Flashrom
  - SelectiveIntellect BootJack (Minimally DARPA CFT funded)
  - Raytheon Pikewerks Firmware Forensics (also CFT-funded, believed to be abandoned)
- McAfee DeepDefender >= 1.6.0
  - Hashes pre-defined files of a UEFI firmware filesystem

Copernicus to the rescue



Simple & Small: 2 Capabilities

- BIOS-writability report
  - "Are we vulnerable to attack?"
  - Indicates whether your BIOS access controls are not properly set, and therefore the systems can be trivially bricked or backdoored
- Integrity report
  - "Have we already been attacked?"
  - Dump BIOS flash chip, compare against all the other machines of the same Vendor/Model/BIOSRevision, or compare against a single known good

Example writability report

(made up numbers)

COUNT	BIOS_VENDOR	PRODUCT_NAME	BIOS_VERSION	BIOS_UNLOCKED
7	Dell Inc.	Latitude E6400	A25	7
3	Dell Inc.	Latitude E6400	A27	3
10	Dell Inc.	Latitude E6400	A29	10
7	Dell Inc.	Latitude E6400	A30	0
1	Dell Inc.	Latitude E6400	A31	0
2	Dell Inc.	Latitude E6400	A32	0
1	Dell Inc.	Latitude E6400	A33	0

This shows that if you update to A30 or newer, you're protected

- Multiple organizations' data (~10k hosts) indicates on average about 55% of machines have unlocked BIOSes!
  - *This means without any special vendor-specific knowledge, an attacker could turn off half your machines and they would never turn back on!*

It might actually be much worse…

COUNT	BIOS_VENDOR	PRODUCT_NAME	BIOS_VERSION	BIOS_UNLOCKED
7	Dell Inc.	Latitude E6400	A25	7
3	Dell Inc.	Latitude E6400	A27	3
10	Dell Inc.	Latitude E6400	A29	10
7	Dell Inc.	Latitude E6400	A30	7
1	Dell Inc.	Latitude E6400	A31	1
2	Dell Inc.	Latitude E6400	A32	2
1	Dell Inc.	Latitude E6400	A33	1
0	Dell Inc.	Latitude E6400	A34	0

Patch released with our help.

- Even if the vendor sets access controls properly, the firmware can have exploitable bugs, just like any other software.
- In the case of the E6400 (and 21 other Dell models) there was a buffer overflow that can allow an attacker to break in.
- We found the bug, and performed responsible disclosure to work with Dell to fix the issue and release a new patch.
- But who patches their BIOSes?
  - *YOU* better start thinking about it! Ounce of prevention >= pound of cure…

But it can be made better

COUNT	BIOS_VENDOR	PRODUCT_NAME	BIOS_VERSION	BIOS_UNLOCKED
31	Dell Inc.	Latitude E6400	A34	0

MITRE applied updates

- MITRE applied the patches once they were available, and has started to incorporate firmware patch management into its standard process
- Copernicus can provided vulnerability situational awareness and configuration management capabilities

BIOS/SMRAM Writability Analysis Demo

- protections.py
- http://youtu.be/wVulh2ADsT4

```
C:\copernicus>python protections.py per-version csv
COUNT BIOS_VENDOR PRODUCT_NAME    BIOS_VERSION SMRAM_UNLOCKED BIOS_UNLOCKED
2     Dell Inc.   Latitude E6430 A11          0             0
1     Dell Inc.   Latitude E6430 A12          0             0

3 CSV files successfully processed
0 (0.0%) CSV files showing unlocked SMRAM
0 (0.0%) CSV files showing unlocked BIOS
0 (out of 0 -- 0.0%) CSV files showing vulnerability to CERT VU#912156
0 (0.0%) CSV files showing vulnerability to CERT VU#255726
0 (0.0%) CSV files showing SMI_LOCK not set
```

0/1, no/yes, can someone easily escalate from ring 0 to SMM, or BIOS?

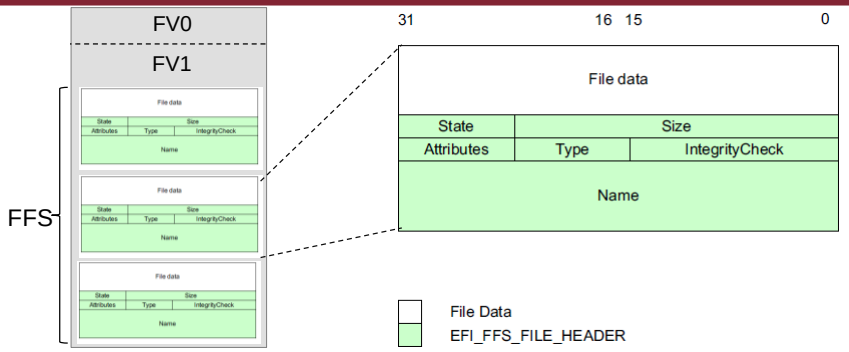
Example CONOPS: Enterprise Situational Awareness

- Deploy Copernicus to endpoints with your typical patch management software, or software deployment mechanism
- Copernicus package includes a script to send output results back to a central server
- Central server runs protections.py once a month to create CSV output.
- CSV output is then emailed to internal security & IT departments to give them visibility into how they're doing on BIOS patch management and how many systems are still vulnerable

Integrity Report

- Modern UEFI BIOSes have a standardized "Firmware FileSystem" (FFS)
  - In contract to vendor-proprietary ways of composing BIOS binary blobs which exist in legacy BIOS
- FFS can be parsed to extract the individual files
  - Files often use the same Portable Executable (PE) format as Windows executables!
    - Sometimes use "Terse Executable" (TE) which is just PE but with smaller headers
- We parse the FFS for two files which purport to be from the same Vendor/Model/Revision and store the results to the OS filesystem.
- Then we uses pair-wise file hashing. If a hash differs, then we can do byte-wise diff, and also parse PE headers to pull out more semantically meaningful information for an analyst

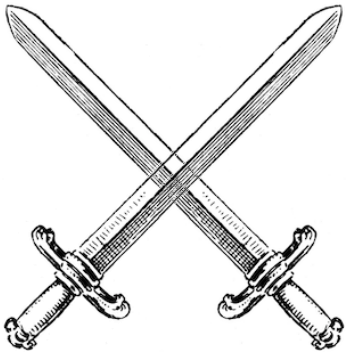
Firmware File System (FFS)



- Firmware Volumes are organized into a FFS
- The base unit of a FFS is a file
- Files can be further subdivided into sections
- Some of the sections will be PE/TE files

Double Edged Sword

- A standardize FFS makes it easier for attackers to decompose BIOS for analysis and finding vulnerabilities or inserting backdoors
- But it's also easier for defenders to analyze the integrity of the BIOS



Demo: Using UEFI Tool to Parse FFS

Structure

Name	Action	Type	Subtype	Text
Intel image		Image	Intel	
Descriptor region		Region	Descriptor	
GbE region		Region	GbE	
ME region		Region	ME	
BIOS region		Region	BIOS	
7A9354D9-0468-444A-81C…		Volume		
7A9354D9-0468-444A-81C…		Volume		
Padding		Padding		
7A9354D9-0468-444A-81C…		Volume		
7A9354D9-0468-444A-81C…		Volume	Boot	
3B42EF57-16D3-44CB-8…		File	PEI module	MemoryInit
CA9D8617-D652-403B-B…		File	PEI module	TxtPei
0D1ED2F7-E92B-4562-9…		File	PEI module	CRBPEI
A27E7C62-249F-4B7B-B…		File	PEI module	DellFlashUpdatePei
1D88C542-9DF7-424A-A…		File	PEI module	WdtPei
92685943-D810-47FF-A…		File	PEI core	CORE_PEI
01359D99-9446-456D-A…		File	PEI module	CpuInitPei
C866BD71-7C79-4BF1-A…		File	PEI module	CpuS3Peim
8B8214F9-4ADB-47DD-A…		File	PEI module	SmmBasePeim
0AC2D35D-1C77-1033-A…		File	PEI module	CpuPolicyPei
1555ACF3-BD07-4685-B…		File	PEI module	CpuPeiBeforeMem
2BB5AFA9-FF33-417B-8…		File	PEI module	CpuPei
C1FBD624-27EA-40D1-A…		File	PEI module	SBPEI
333BB2A3-4F20-4C8B-A…		File	PEI module	AcpiPlatformPei
0F69F6D7-0E4B-43A6-B…		File	PEI module	WdtAppPei

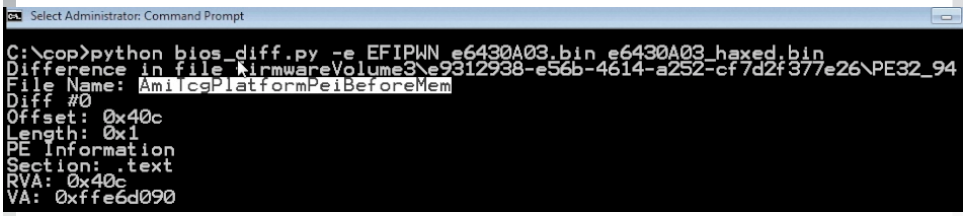
Information

```
FileSystem GUID:
7A9354D9-0468-444A-81C
E-0BF617D890DF
Size: 00200000
Revision: 1
Attributes: ffff8eff
Erase polarity: 1
Header size: 0048
```

UEFI Tool: https://github.com/LongSoft/UEFITool

BIOS Change Detection Demo

- bios_diff.py
- http://www.youtube.com/watch?v=XaeMrL1LqPo



Output or no output, are there any unexpected changes to the BIOS?

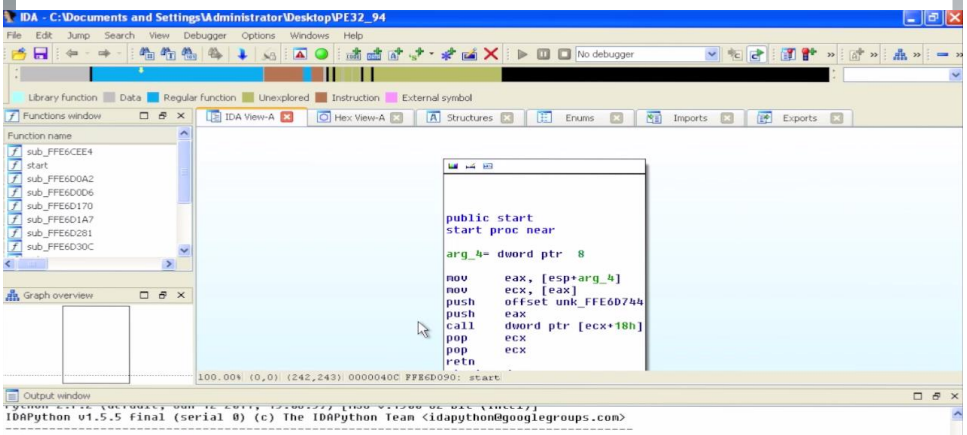
Example CONOPS:
International Evil Maid (of Mystery!) Detection

- Run Copernicus on loaner laptops before they are taken overseas for travel
- Burn BIOS dump to CD
- When person returns, re-run Copernicus, and run bios_diff.py on the copy from the HD and the copy from the CD and see if they differ
- If so, forward bios_diff.py output to malware analyst

Example CONOPS: Enterprise Situational Awareness

- Deploy Copernicus to endpoints with your typical patch management software, or software deployment mechanism
- Copernicus package includes a script to send output results back to a central server
- Central server runs bios_diff.py on each new BIOS as it comes in.
- If it passes the checks it's thrown away, if it doesn't pass, the output and two files are archived and sent to a malware analyst

Demo: Making sense of UEFI PE files in IDA Pro



Where can I get Copernicus today?

- Copernicus has been run on ~10k Windows 7 endpoints. (All of MITRE's + some other organizations')
- Available as a free binary-only download
  - We make the src available to people that are willing to contribute data back to us
  - http://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/copernicus-question-your-assumptions-about or just google "MITRE Copernicus"

Where can I get Copernicus tomorrow?

- Tell Mandiant (and the other security vendors you may use) that you want visibility at the BIOS level!
- We can then work out a deal with them where they get the source code for free, and we get back some data to help our ongoing research.
- It's a win for all parties

Questions?

- Thanks for listening!

- Email contact:

xkovah, ckallenberg, jbutterworth, scornwell at mitre dot org

- Twitter contact:

@xenokovah, @coreykal, @jwbutterworth3, @ssc0rnwell

p.s. If you don't already know about it, go check out OpenSecurityTraining.info!

Copernicus Caveats

- Only Intel CPU/chipset support, no AMD support
  - We'll add AMD when someone says they have a lot of those machines and they're willing to contribute data back to us
- Only supports Windows 7 32 & 64 bit and newer
  - Doesn't *officially* support Windows 8 but it's been known to run on it for some people, and not for others
  - Adding support for XP and greater, but mainly because we want Win2k3 support and they share a kernel.
- Bios_diff.py doesn't diff UEFI variables yet
  - It's on our todo list
- Fundamentally untrustworthy…a kernel mode attacker can make it lie…just like every other piece of security software you currently use
  - Copernicus 2 is *much* more trustworthy, but it requires a TPM (with the requisite secure provisioning), and CPU support for Intel TXT, but also doesn't support 64 bit yet
    - Copernicus 3 will be even better :)

References

- [1] Attacking Intel BIOS – Alexander Tereshkin & Rafal Wojtczuk – Jul. 2009 http://invisiblethingslab.com/resources/bh09usa/Attacking%20Intel%20BIOS.pdf
- [2] TPM PC Client Specification - Feb. 2013 http://www.trustedcomputinggroup.org/developers/pc_client/specifications/
- [3] Evil Maid Just Got Angrier: Why Full-Disk Encryption With TPM is Insecure on Many Systems – Yuriy Bulygin – Mar. 2013 http://cansecwest.com/slides/2013/Evil%20Maid%20Just%20Got%20Angrier.pdf
- [4] A Tale of One Software Bypass of Windows 8 Secure Boot – Yuriy Bulygin – Jul. 2013 http://blackhat.com/us-13/briefings.html#Bulygin
- [5] Attacking Intel Trusted Execution Technology - Rafal Wojtczuk and Joanna Rutkowska – Feb. 2009 http://invisiblethingslab.com/resources/bh09dc/Attacking%20Intel%20TXT%20-%20paper.pdf
- [6] Another Way to Circumvent Intel® Trusted Execution Technology - Rafal Wojtczuk, Joanna Rutkowska, and Alexander Tereshkin – Dec. 2009 http://invisiblethingslab.com/resources/misc09/Another%20TXT%20Attack.pdf
- [7] Exploring new lands on Intel CPUs (SINIT code execution hijacking) - Rafal Wojtczuk and Joanna Rutkowska – Dec. 2011 http://www.invisiblethingslab.com/resources/2011/Attacking_Intel_TXT_via_SINIT_hijacking.pdf
- [7] Meet 'Rakshasa,' The Malware Infection Designed To Be Undetectable And Incurable - http://www.forbes.com/sites/andygreenberg/2012/07/26/meet-rakshasa-the-malware-infection-designed-to-be-undetectable-and-incurable/

References 2

- [8] Implementing and Detecting an ACPI BIOS Rootkit – Heasman, Feb. 2006 http://www.blackhat.com/presentations/bh-europe-06/bh-eu-06-Heasman.pdf
- [9] Implementing and Detecting a PCI Rookit – Heasman, Feb. 2007 http://www.blackhat.com/presentations/bh-dc-07/Heasman/Paper/bh-dc-07-Heasman-WP.pdf
- [10] Using CPU System Management Mode to Circumvent Operating System Security Functions - Duflot et al., Mar. 2006 http://www.ssi.gouv.fr/archive/fr/sciences/fichiers/lti/cansecwest2006-duflot-paper.pdf
- [11] Getting into the SMRAM:SMM Reloaded – Duflot et. Al, Mar. 2009 http://cansecwest.com/csw09/csw09-duflot.pdf
- [12] Attacking SMM Memory via Intel® CPU Cache Poisoning – Wojtczuk & Rutkowska, Mar. 2009 http://invisiblethingslab.com/resources/misc09/smm_cache_fun.pdf
- [13] Defeating Signed BIOS Enforcement – Kallenberg et al., Sept. 2013
- http://www.syscan.org/index.php/download/get/6e597f6067493dd581eed737146f3afb/SySCan2014_CoreyKallenberg_SetupforFailureDefeatingSecureBoot.zip
- [14] Mebromi: The first BIOS rootkit in the wild – Giuliani, Sept. 2011 http://www.webroot.com/blog/2011/09/13/mebromi-the-first-bios-rootkit-in-the-wild/

References 3

- [15] Persistent BIOS Infection – Sacco & Ortega, Mar. 2009 http://cansecwest.com/csw09/csw09-sacco-ortega.pdf
- [16] Deactivate the Rootkit – Ortega & Sacco, Jul. 2009 http://www.blackhat.com/presentations/bh-usa-09/ORTEGA/BHUSA09-Ortega-DeactivateRootkit-PAPER.pdf
- [17] Sticky Fingers & KBC Custom Shop – Gazet, Jun. 2011 http://esec-lab.sogeti.com/dotclear/public/publications/11-recon-stickyfingers_slides.pdf
- [18] BIOS Chronomancy: Fixing the Core Root of Trust for Measurement – Butterworth et al., May 2013 http://www.nosuchcon.org/talks/D2_01_Butterworth_BIOS_Chronomancy.pdf
- [19] New Results for Timing-based Attestation – Kovah et al., May 2012 http://www.ieee-security.org/TC/SP2012/papers/4681a239.pdf

References 4

- [20] Low Down and Dirty: Anti-forensic Rootkits - Darren Bilby, Oct.2006 http://www.blackhat.com/presentations/bh-jp-06/BH-JP-06-Bilby-up.pdf
- [21] Implementation and Implications of a Stealth Hard-Drive Backdoor – Zaddach et al., Dec. 2013 https://www.ibr.cs.tu-bs.de/users/kurmus/papers/acsac13.pdf
- [22] Hard Disk Hacking – Sprite, Jul. 2013http://spritesmods.com/?art=hddhack
- [23] Embedded Devices Security and Firmware Reverse Engineering - Zaddach & Costin, Jul. 2013 https://media.blackhat.com/us-13/US-13-Zaddach-Workshop-on-Embedded-Devices-Security-and-Firmware-Reverse-Engineering-WP.pdf
- [24] Can You Still Trust Your Network Card – Duflot et al., Mar. 2010 http://www.ssi.gouv.fr/IMG/pdf/csw-trustnetworkcard.pdf
- [25] Project Maux Mk.II, Arrigo Triulzi, Mar. 2008 http://www.alchemistowl.org/arrigo/Papers/Arrigo-Triulzi-PACSEC08-Project-Maux-II.pdf

References 5

- [26] Copernicus: Question your assumptions about BIOS Security – Butterworth, July 2013 http://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/copernicus-question-your-assumptions-about
- [27] Copernicus 2: SENTER the Dragon – Kovah et al., Mar 2014 https://cansecwest.com/slides/2014/Copernicus2-SENTER_the-Dragon-CSW.pptx
- [28] Playing Hide and Seek with BIOS Implants – Kovah, Mar 2014 http://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/playing-hide-and-seek-with-bios-implants
- [29] Setup for Failure: Defeating UEFI – Kallenberg et al., Apr 2014 http://syscan.org/index.php/download/get/6e597f6067493dd581eed737146f3afb/SyScan2014_CoreyKallenberg_SetupforFailureDefeatingSecureBoot.zip
- [30] SENTER Sandman: Using Intel TXT to Attack BIOSes – Kovah et al., June 2014
  slides not posted anywhere yet
- [31] 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