presented by

SMM Protection in EDK II

Spring 2017 UEFI Seminar and Plugfest

March 27 - 31, 2017

Jiewen Yao, Intel Corporation

Agenda

- Known SMM Attacks
- More Protection
  - SMM Memory Protection
  - CommBuffer Enforcement
  - ASLR in SMM
  - Guard Page
  - Reduce SMI Handler
- Summary / Call to Action

What is SMM and SMI?

- System Management Mode (SMM)
  - Is a special CPU operating mode.
  - Is inside of a special SMM memory (SMRAM)
  - Access the whole system memory and IO, including OS memory and hypervisor memory.
  - Is invoked through a System Management Interrupt (SMI)
  - Has software executive (SMI handler) to perform operation based upon different SMI.

Known SMM Attacks

SMM Attack	Description	Example
SMRAM is unlocked	An attacker can set register to unlock SMRAM, and override SMRAM.	A.1, A.2
Cache Poisoning	An attacker can set CPU cache to override SMRAM.	A.3, A.4
SMRAM remap	An attacker can control chipset register to remap a normal system memory to SMRAM.	A.5
Branch Outside of SMRAM	SMM code calls outside of SMRAM, which is controlled by the attacker.	A.6, A.7
SMM Communication Buffer Attack	SMM code uses SMM communication buffer to exchange information with non-SMM agent. The attacker can give a malicious communication buffer to SMM, and the SMM may write SMRAM or Virtual Machine Monitor (VMM).	A.8, A.9

Known Mitigation

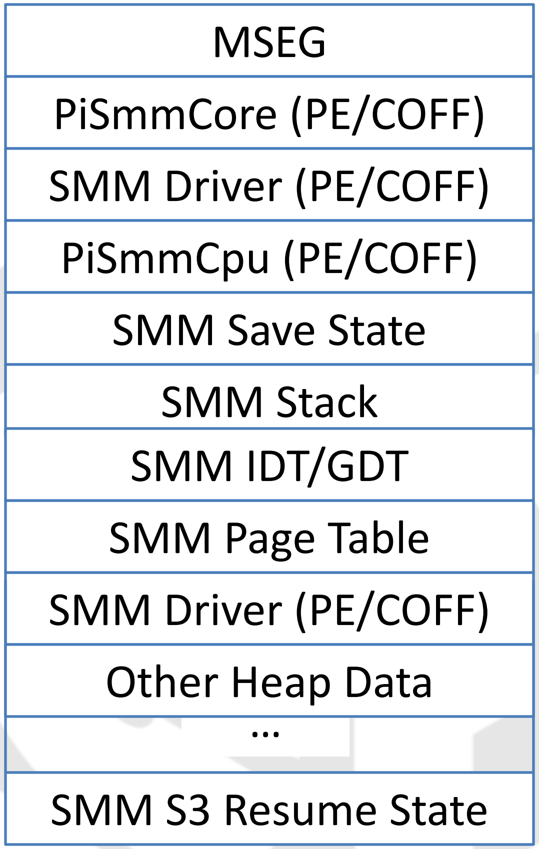
SMM Attack	Mitigation
SMRAM is unlocked	1) Lock SMRAM at PI SmmReadyToLock.
Cache Poisoning	1) Enable SMM Range Register.
SMRAM remap	1) Lock Remap register.
Branch Outside of SMRAM	1) Enable Smm_Code_Access Register. 2) Setup Non-Executable (NX) paging outside of SMM.
SMM Communication Attack	1) Check SMM Communication Buffer. 2) Check MemoyMapped IO (MMIO) bar access.

New methods may be discovered

SMM Memory Protection

Current SMRAM Layout

- Every page in SMRAM is read/write
- Every page in SMRAM is executable



MSEG

PiSmmCore (PE/COFF)

SMM Driver (PE/COFF)

PiSmmCpu (PE/COFF)

SMM Save State

SMM Stack

SMM IDT/GDT

SMM Page Table

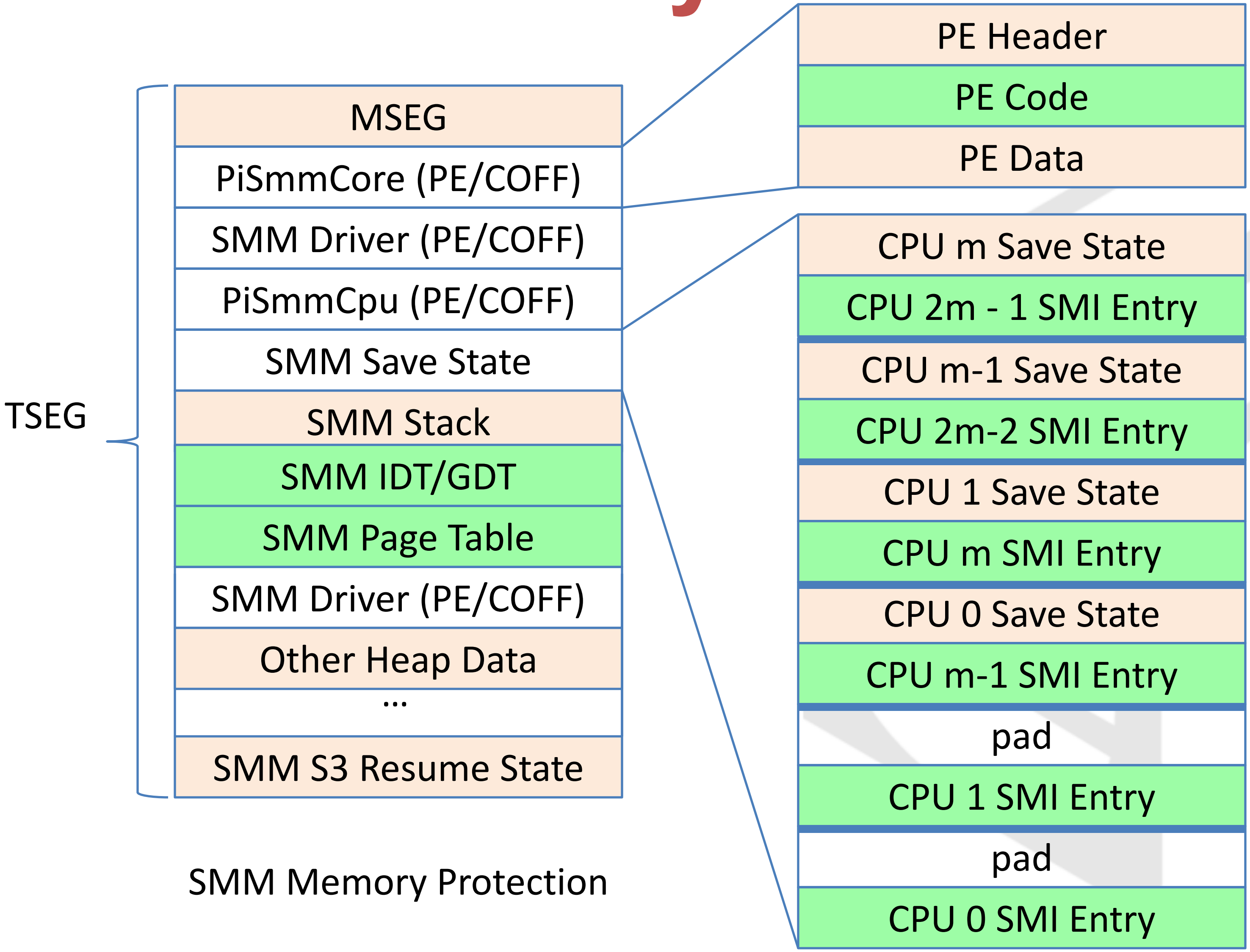
SMM Driver (PE/COFF)

Other Heap Data

…

SMM S3 Resume State

SMM Memory Protection



TSEG

MSEG

PiSmmCore (PE/COFF)

SMM Driver (PE/COFF)

PiSmmCpu (PE/COFF)

SMM Save State

SMM Stack

SMM IDT/GDT

SMM Page Table

SMM Driver (PE/COFF)

Other Heap Data

…

SMM S3 Resume State

PE Header

PE Code

PE Data

CPU m Save State

CPU 2m - 1 SMI Entry

CPU m-1 Save State

CPU 2m-2 SMI Entry

CPU 1 Save State

CPU m SMI Entry

CPU 0 Save State

CPU m-1 SMI Entry

pad

CPU 1 SMI Entry

pad

CPU 0 SMI Entry

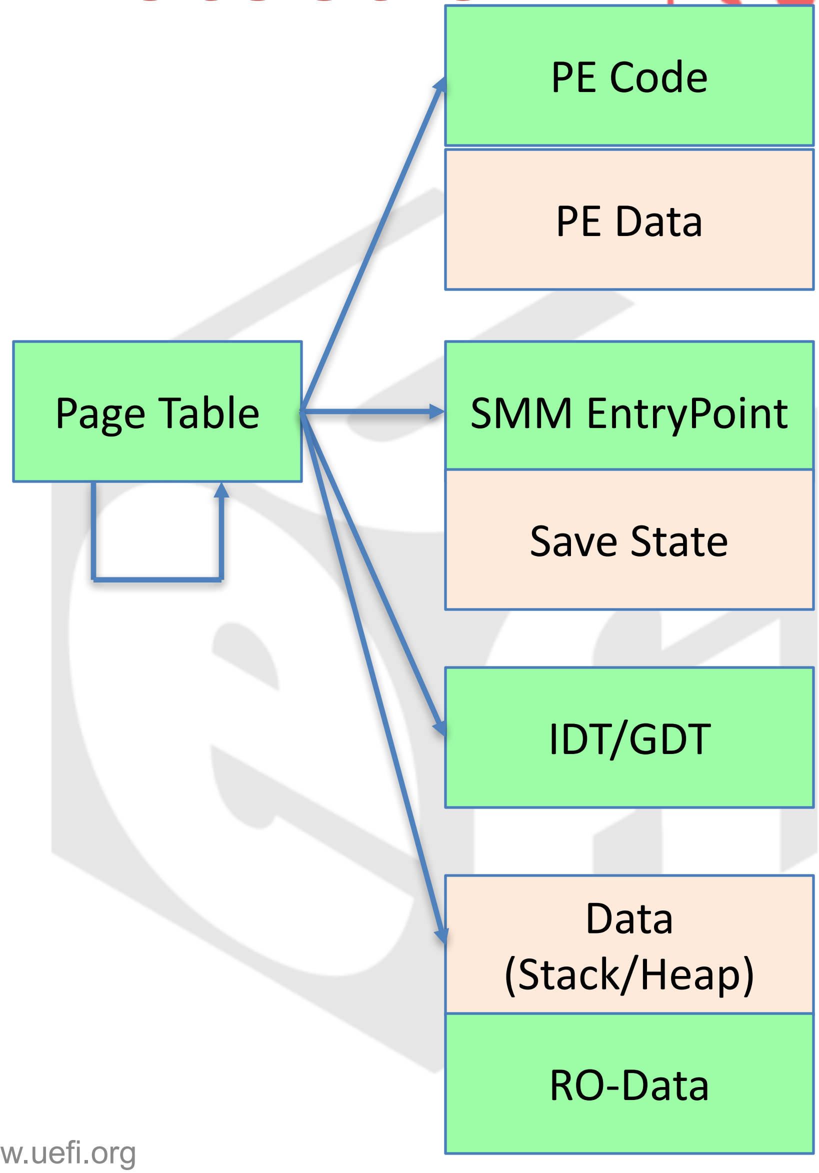
SMM Memory Protection

🟩	RO
▨	XD

SMM Memory Protection

- Using static page table
- Set NonExecutable (NX) for data
- Set ReadOnly(RO) for code
- Protect page itself

- SMM driver can protect its own critical data in ReadOnly(RO) memory



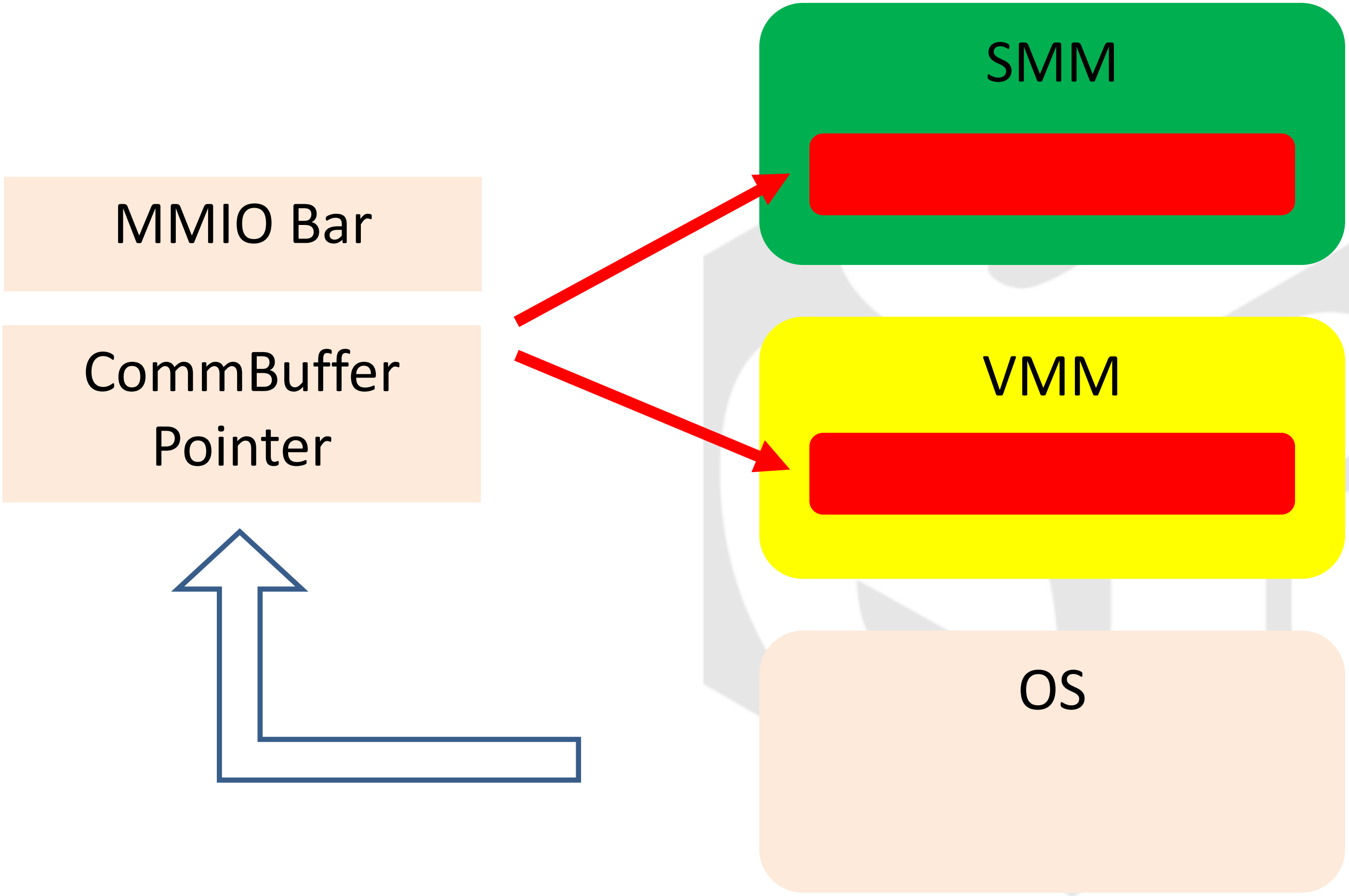
SMM Memory Protection

- Prevents code injection
- Protects critical data (read-only)

- Limitations
  - Return-oriented programming (ROP) attack.
  - Size overhead
    - PE image: 6K * SmmImageCount (average)
    - Static Page Table: 2M (1G paging for 48bit)

CommBuffer Enforcement

SMM CommBuffer Attack



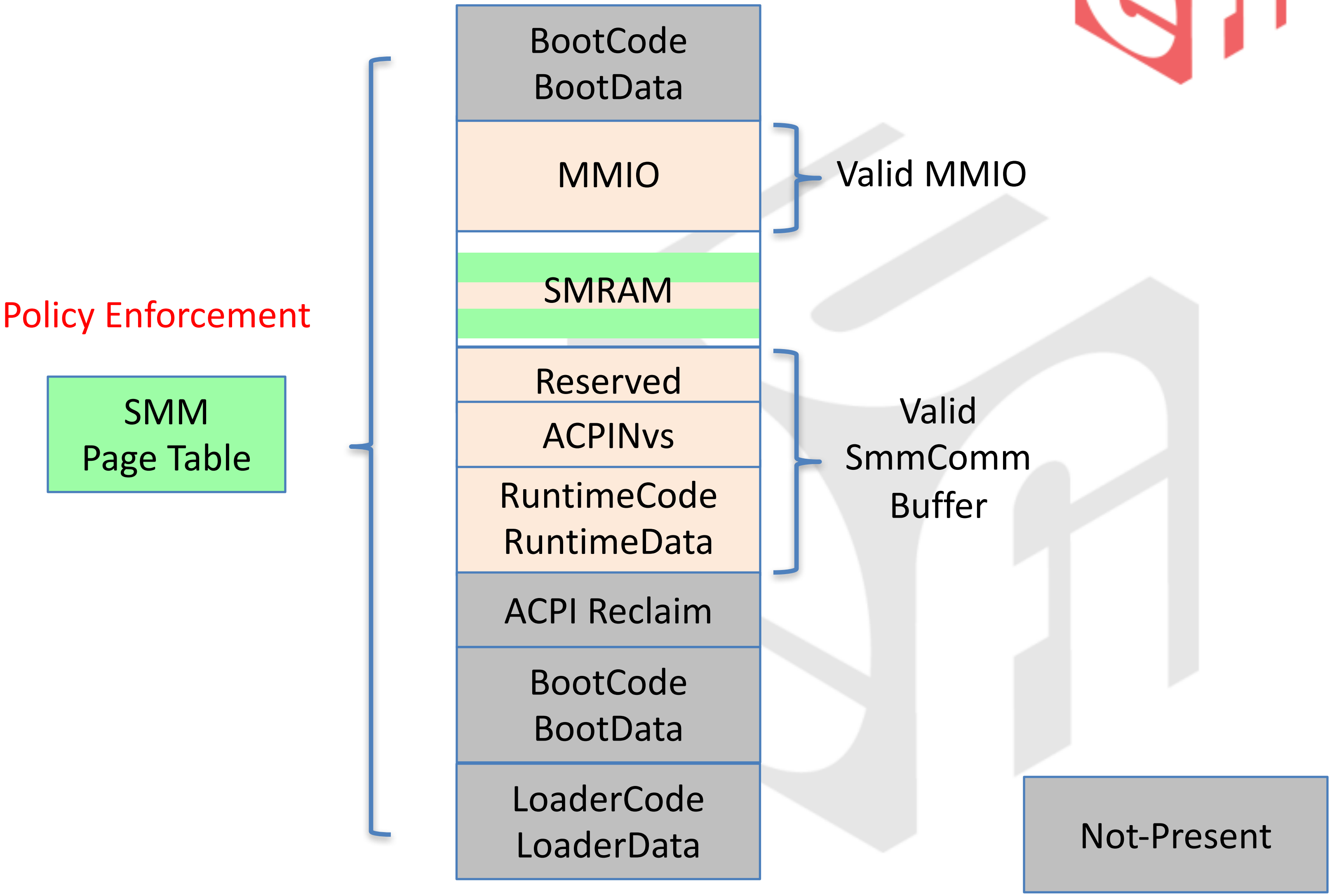
Current CommBuffer Check

- SMI handler MUST check SMM communication buffer content by writing code like below:

```
if (!SmmIsBufferOutsideSmmValid ((UINTN)CommBuffer, TempCommBufferSize)) {
  DEBUG ((EFI_D_ERROR, "SmmVariableHandler: SMM communication buffer in 
SMRAM or overflow!\n"));
  return EFI_SUCCESS;
}
```

- But if the check is missing, there is no way to detect such problem.

CommBuffer Enforcement



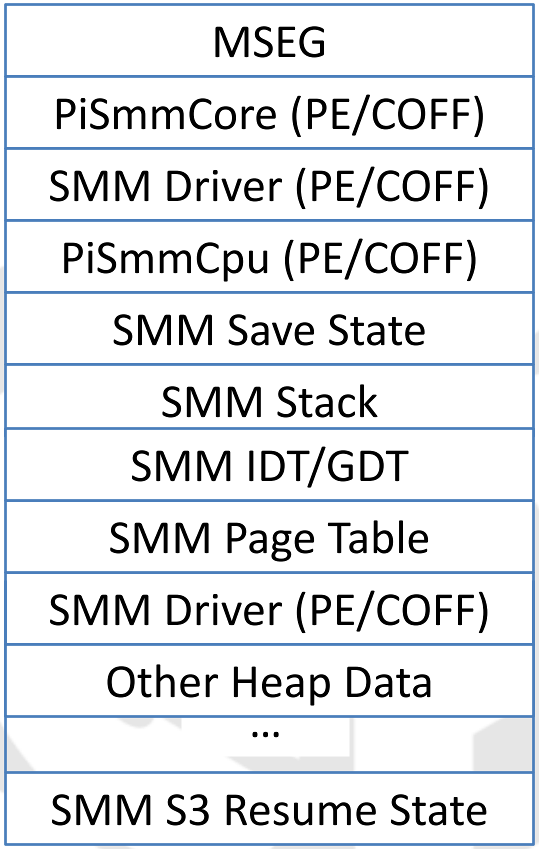
CommBuffer Enforcement

- Resist comm buffer attack even the CommBuffer check is missing in SMM driver
- Protects hypervisors
- Limitation
  - This enforcement is not applied to hotplug memory, which is still read/write.
  - MMIO region is mapped. SMI handler need make sure MMIO bar point to a valid region.

Address Space Layout Randomization (ASLR) in SMM

Current SMM Layout

- The layout is fixed.
- This attack may find out a sequence of instruction existed in code region (gadgets) and execute. (ROP)
- This ROP attack can bypass NX/RO protection.



MSEG

PiSmmCore (PE/COFF)

SMM Driver (PE/COFF)

PiSmmCpu (PE/COFF)

SMM Save State

SMM Stack

SMM IDT/GDT

SMM Page Table

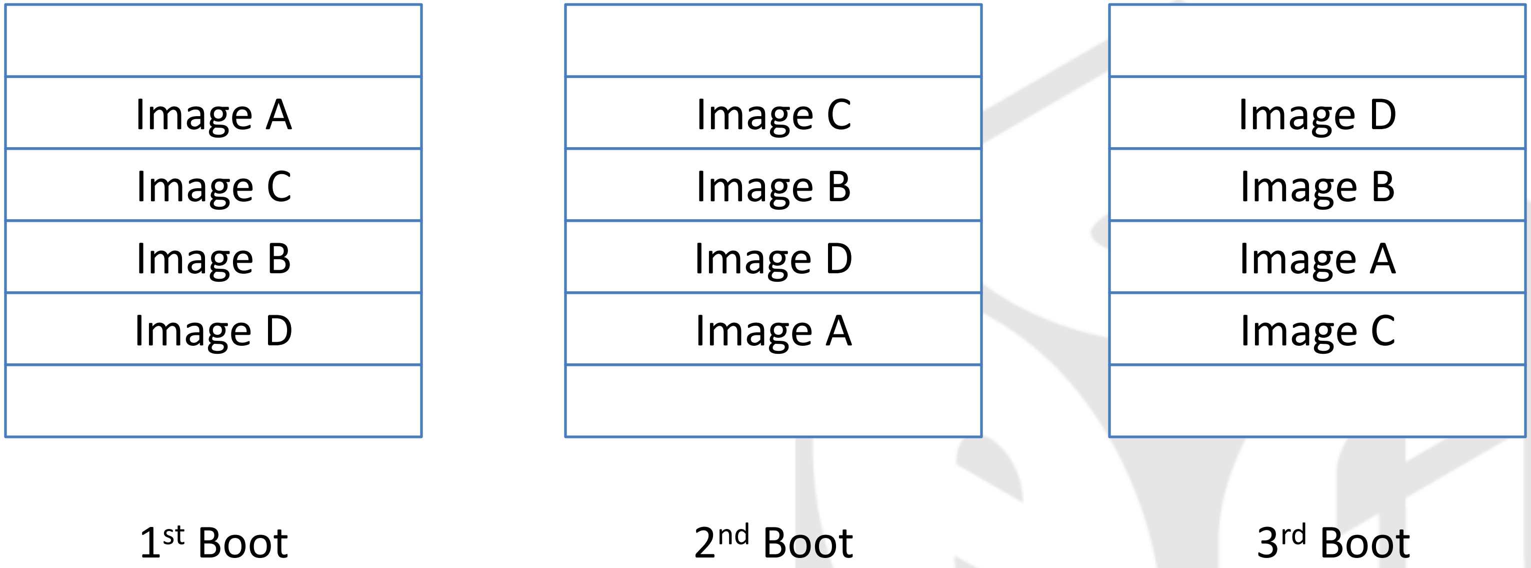
SMM Driver (PE/COFF)

Other Heap Data

…

SMM S3 Resume State

Image Shuffle

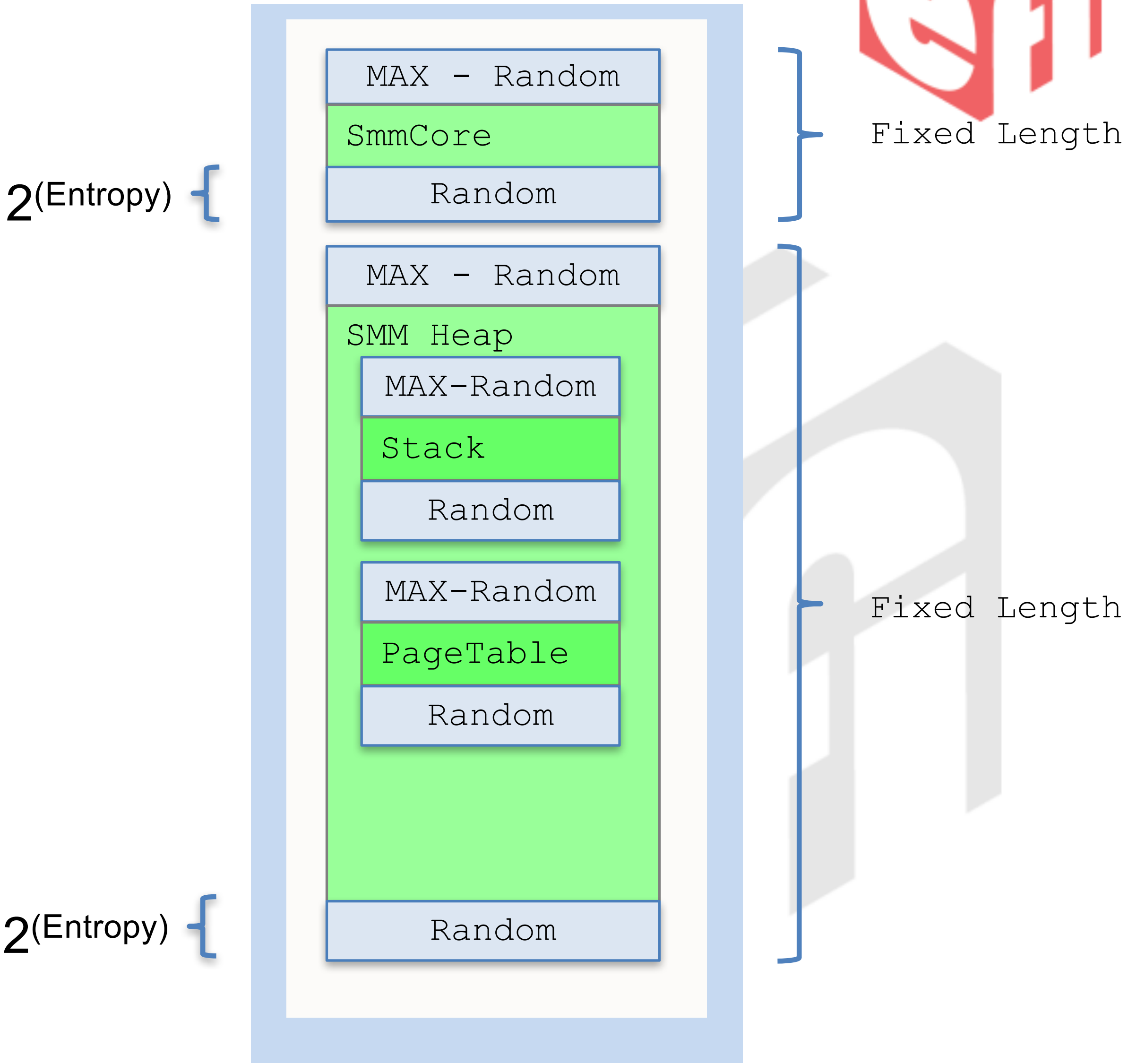


1st Boot	2nd Boot	3rd Boot
Image A	Image C	Image D
Image C	Image B	Image B
Image B	Image D	Image A
Image D	Image A	Image C

As such, it makes difficult for attacker to locate gadgets for ROP attack.

Heap Shift

It makes difficult for attacker to locate gadget for ROP attack.



ASLR in SMM

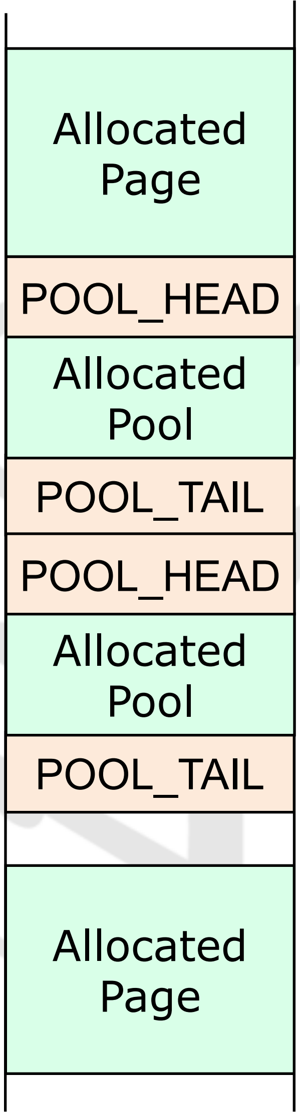
- Make Buffer Overflow/ROP attack harder, because the memory layout is changed in each boot.

- Limitations
  - – SMM is a resource constrained environment. Entropy for Heap Shift might be not so big.
  - – Information leakage in SMM (LoadedImageProtocol)

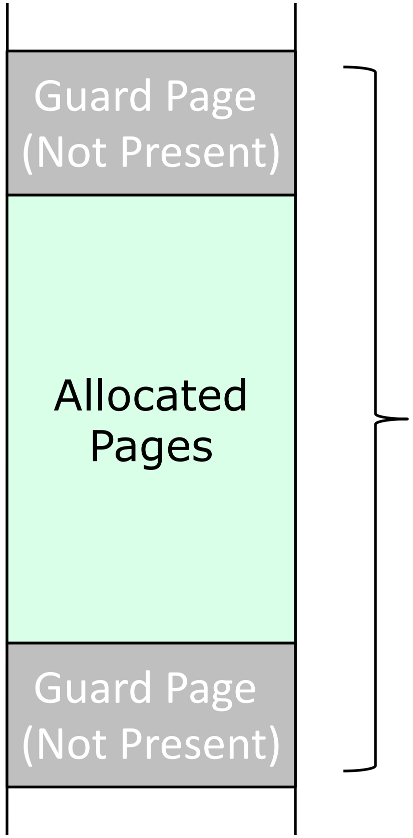
Guard Page

Current Memory Allocation

- Page overflow cannot be detected
- Pool overflow can only be detected when memory is freed, because of POOL_TAIL signature check at FreePool()



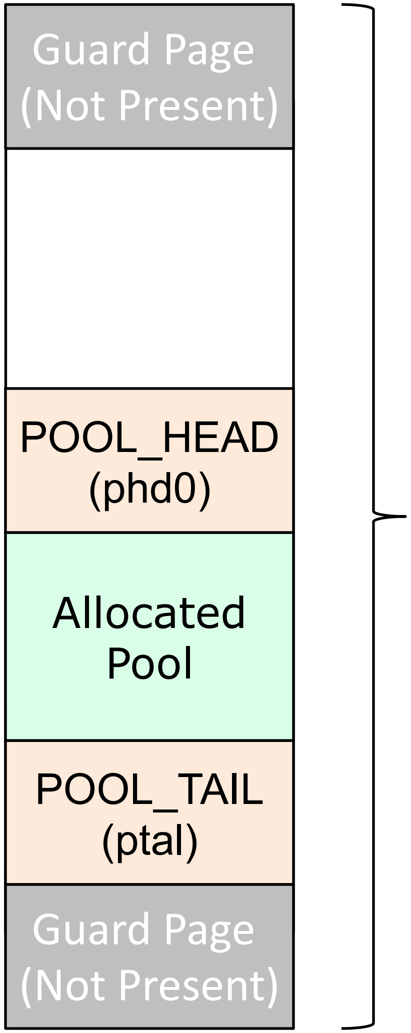
New Page Allocation



One Allocation for AllocatePages()

2 guard pages (8K)

New pool allocation



One Allocation for
AllocatePool()

2 guard pages (8K)
+ 4K page alignment

Guard Page

- Catch page overflows when they happen
- Catch pool overflows when they happen

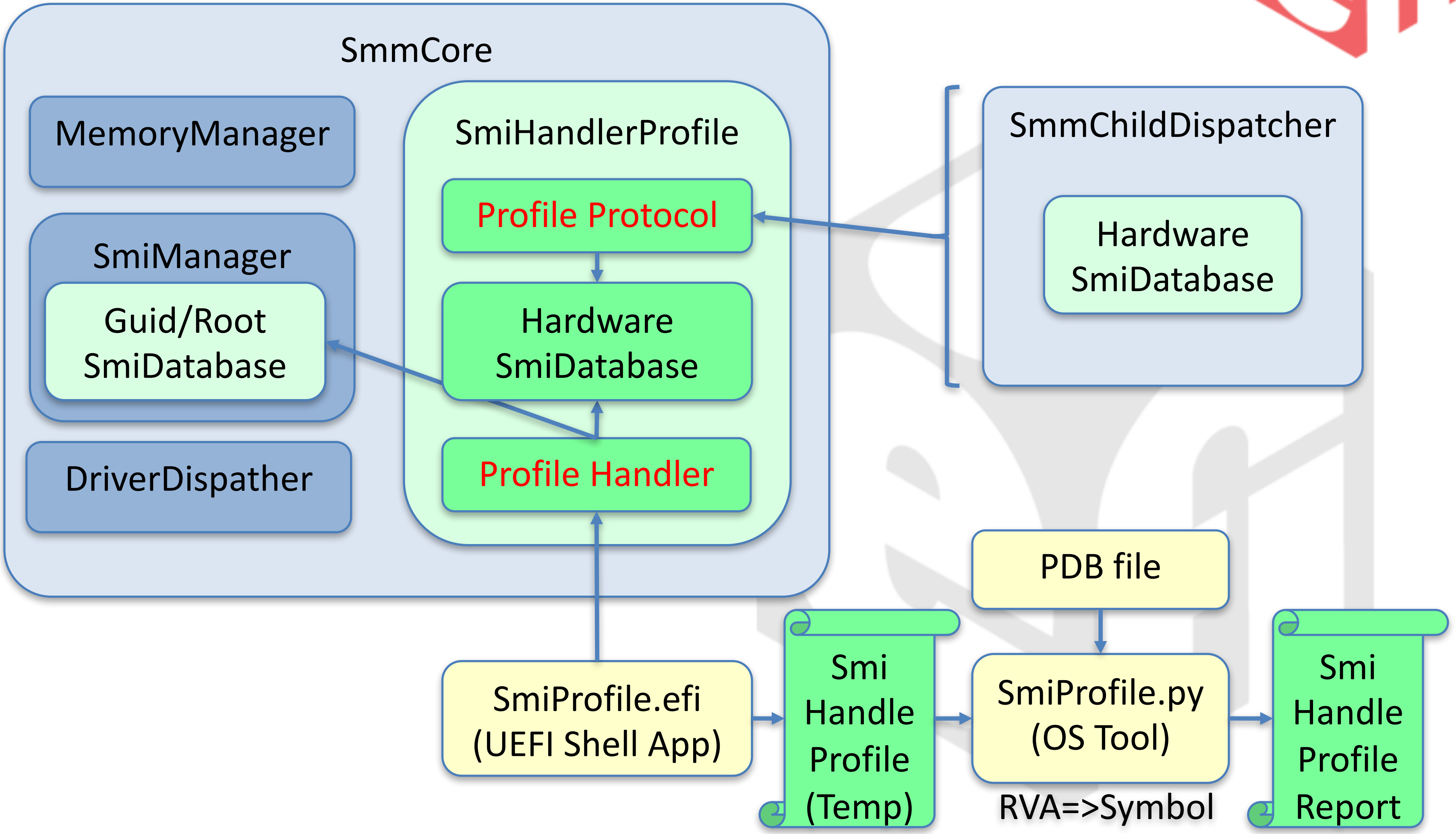
- Limitation
  - Memory size overhead
    - Additional 8K for each page allocation.
    - Additional 8K+4K alignment for each pool allocation.
    - It might need above 128M SMRAM.
  - A debug feature, because of size overhead.

Reduce SMI Handler

SMI Handlers

- SMI Handler == Attack Surface
- Question:
  - How many SMI handlers in the BIOS?
  - How many Root SMI handlers, GUID handlers, software SMI handlers, …… ?

SMI Handler Profile



SMI Handler Profile

- Developer can check if the SMI handler is necessary
- Test engineer can use it for validation

- Limitation
  - Only used as a debug feature (info leakage)
  - The profile only shows info, which requires further analysis

Summary

- SMM is a target due to high execution privilege

- There are known SMM attacks and mitigations

- Developers can do more to protect SMM
  - SMM Memory Protection
  - CommBuffer Enforcement
  - ASLR in SMM
  - Guard Page
  - Reduce Number of SMI Handlers

Call To Action

- Adopt “SMM Memory Protection” and “CommBuffer enforcement” to harden the platform. [P.1][P.2]
- Use “SMI handler profile” to audit the SMI handlers. [P.3]
- Evaluate “ASLR in SMM” and resolve information leakage. [P.4]
- Use “GuardPage” to validate buffer overflow. [P.4]

Acknowledgement

- Some content of the material is discussed with UEFI BIOS and security experts

- Special thanks to Vincent Zimmer (Intel), Kirt Brannock (Intel), Jeremiah Cox (Microsoft), Sean Brogan (Microsoft)

Reference

- Attacks
  - [A.1] Using CPU SMM to Circumvent OS Security Functions (http://fawlty.cs.usfca.edu/~cruse/cs630f06/duflot.pdf)
  - [A.2] Using SMM For Other Purposes (http://phrack.org/issues/65/7.html)
  - [A.3] Attacking SMM Memory via Intel Cache Poisoning (http://invisiblethingslab.com/resources/misc09/smm_cache_fun.pdf)
  - [A.4] Getting Into the SMRAM: SMM Reloaded (http://www.politicalavenue.com/libraryebooks/cryptology-and-cryptography/csw09-duflot.pdf)
  - [A.5] Attacking-Intel-TXT (http://invisiblethingslab.com/resources/bh09dc/Attacking%20Intel%20TXT%20-%20slides.pdf)
  - [A.6] BIOS SMM Privilege Escalation Vulnerabilities (http://www.securityfocus.com/archive/1/505590)
  - [A.7] System Management Mode Design and Security Issues (http://www.ssi.gouv.fr/uploads/IMG/pdf/IT_Defense_2010_final.pdf)
  - [A.8] A New Class of Vulnerabilities in SMI Handlers (https://cansecwest.com/slides/2015/A%20New%20Class%20of%20Vulnin%20SMI%20-%20Andrew%20Furtak.pdf)
  - [A.9] BARingthe System (http://www.intelsecurity.com/advanced-threat-research/content/data/REConBrussels2017_BARing_the_system.pdf)

Reference

- Protection:
  - [P.1] A_Tour_Beyond_BIOS_Secure_SMM_Communication (https://github.com/tianocore-docs/Docs/raw/master/White_Papers/A_Tour_Beyond_BIOS_Secure_SMM_Communication.pdf)
  - [P.2] A_Tour_Beyond_BIOS_Memory_Protection_in_UEFI (https://www.gitbook.com/book/edk2-docs/a-tour-beyond-bios-memory-protection-in-uefi-bios/details)
  - [P.3] SMI Handler Profile Feature (https://github.com/tianocore/tianocore.github.io/wiki/SMI-handler-profile-feature)
  - [P.4] A_Tour_Beyond_BIOS_Securiy_Enhancement_to_Mitigate_Buffer_Overflow_in_UEFI (https://github.com/tianocore-docs/Docs/raw/master/White_Papers/A_Tour_Beyond_BIOS_Securiy_Enhancement_to_Mitigate_Buffer_Overflow_in_UEFI.pdf)

Thanks for attending the Spring 2017 UEFI Seminar and Plugfest

For more information on the UEFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by