presented by

Microsoft UEFI Updates
Spring 2019 Edition

Spring 2019 UEFI Plugfest
April 8-12, 2019
Presented by Jeremiah Cox (Microsoft)

Agenda

- Kernel DMA Protection
- Modern Servicing
- Project Mu
- Questions?

Windows 10: Kernel DMA Protection

Win10: Kernel DMA Protection

A.k.a. DMA Guard, Memory Access Protection

Goal: Mitigate drive-by DMA attacks!

- End-to-end story depends on UEFI
- Drive-by DMA ports must be blocked or sandboxed

Win10: Kernel DMA Protection

IHV Call to Action

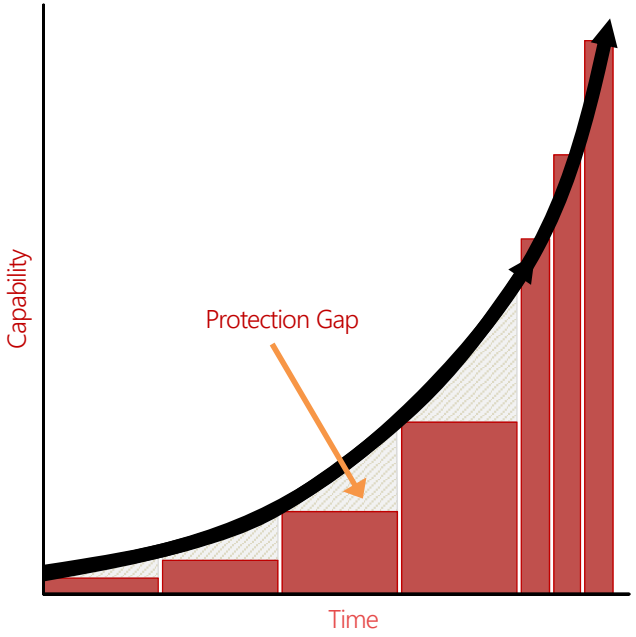
- Support sandboxing
  - Use EFI_PCI_IO_PROTOCOL protocols for DMA
  - Map(), Unmap(), AllocateBuffer(), …

OEM Call to Action

- Read the “Kernel DMA Protection” docs
  - https://docs.microsoft.com/en-us/windows/security/information-protection/kernel-dma-protection-for-thunderbolt
- Check out our tests
  - https://github.com/Microsoft/mu_plus/tree/release/201903/UefiTestingPkg/AuditTests/DMAProtectionAudit/UEFI/VTd
  - https://github.com/Microsoft/mu_plus/tree/release/201903/UefiTestingPkg/AuditTests/DMAProtectionAudit/Windows
- Chat with us

Modern Servicing

What is Modern Servicing?



- Attackers take advantage of periods between releases
- Stay ahead of the attackers with continual software improvements

A multi-year vision to **reduce the total cost of servicing for the ecosystem**, including:

Scalable solution across ecosystem of OEMs, ODMs, IBVs, and IHVs

Consistent **servicing workflow** for all components

Independently updatable software

Shared insights on health

Comprehensive **remediation** (µCode, firmware, drivers, OS, apps)

Modern Servicing: Firmware

Goal: Protect the Ecosystem

Fix firmware security issues…

- Quickly
- Comprehensively
- Reliably

Project Mu

What is Project Mu ?

- Tianocore…
  - Restructured
- OSS example of how Surface achieves Modern Servicing

Project Mu Principles

- 1 Tree
  - All products share identical core code
- Stay current, pull at each EDK2 release
  - Upstream changes into EDK2
- Focus on the 1 tree, not servicing forks
- Less is More

Project Mu: Code Example

```
project_mu_surface_laptop/
├── Build/
├── Conf/
├── MU_BASECORE/
├── Common/
│   ├── MU_TIANO_PLUS/
│   ├── MU_PLUS/
│   ├── MSCORE_INTERNAL/    # Proprietary code and code not yet approved for public distribution
│   └── SURFACE/            # Shared code to enable common features like FrontPage
├── Silicon/
│   ├── Intel/
│   │   ├── MU_SILICON_INTEL_TIANO/       # Project Mu Intel Code from TianoCore
│   │   └── KBL/            # Intel KBL Reference Code
│   └── ...
└── Platform/
    ├── Surface/
    │   ├── SurfKbl/
    │   │   ├── KblFamily/  # Surface Customizations/Overrides for KBL Ref Code
    │   │   └── Laptop/     # Surface Laptop-Specific Platform Code
    │   └── ...
    └── ...
```

Reference: https://microsoft.github.io/mu/WhatAndWhy/layout/#surface-laptop-example

Project Mu: Validation

- Automated test pass
- Builds smoke tested by dev
- WU sends to unfused Canary group
- WU sends to unfused Selfhost group
- Sent to “validation”
- Production signed
- Flight rings, telemetry, …

In Closing

- DMA links:
  - https://docs.microsoft.com/en-us/windows/security/information-protection/kernel-dma-protection-for-thunderbolt
  - https://github.com/Microsoft/mu_plus/tree/release/201903/UefiTestingPkg/AuditTests/DMAProtectionAudit/UEFI/VTd
  - https://github.com/Microsoft/mu_plus/tree/release/201903/UefiTestingPkg/AuditTests/DMAProtectionAudit/Windows
- Chat with us

Thanks for attending the 2019 Spring UEFI Plugfest

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

presented by