Open Source Test Tools for UEFI

Ricardo Neri, Software Engineer
Vincent Zimmer, Sr. Principal Engineer

presented by

intel®

UEFI Plugfest – May 2014

Agenda

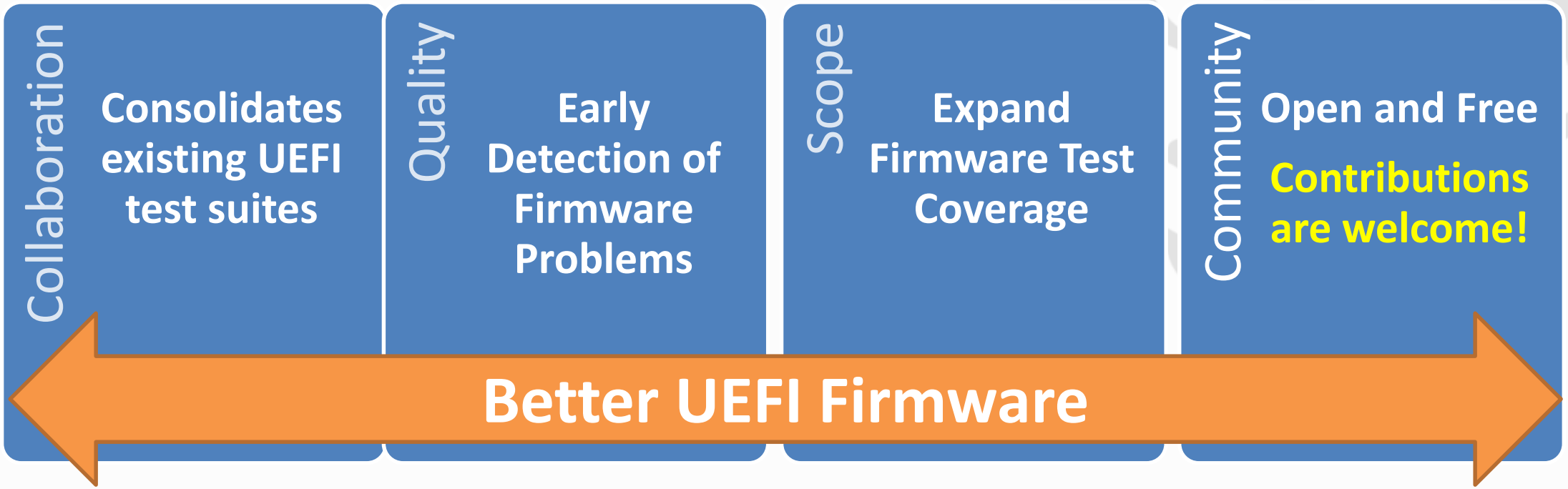
Linux UEFI Validation Operating System

CHIPSEC - Platform Security Assessment Framework

What Now?

Linux UEFI Validation Operating System

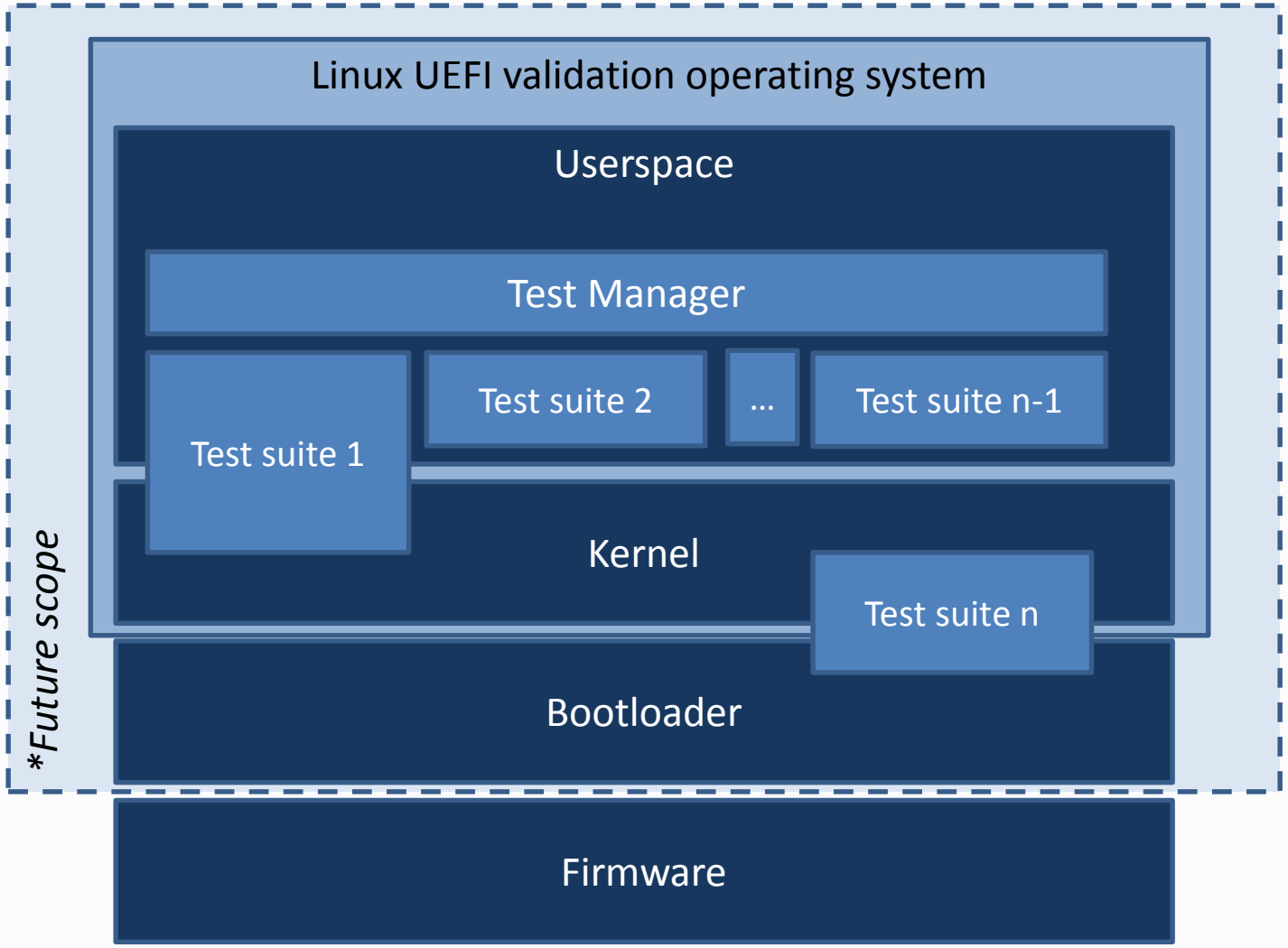
Objective: Create a Linux-readiness validation operating system which helps firmware engineers and Linux contributors reduce development and enabling time for Linux on an UEFI system.



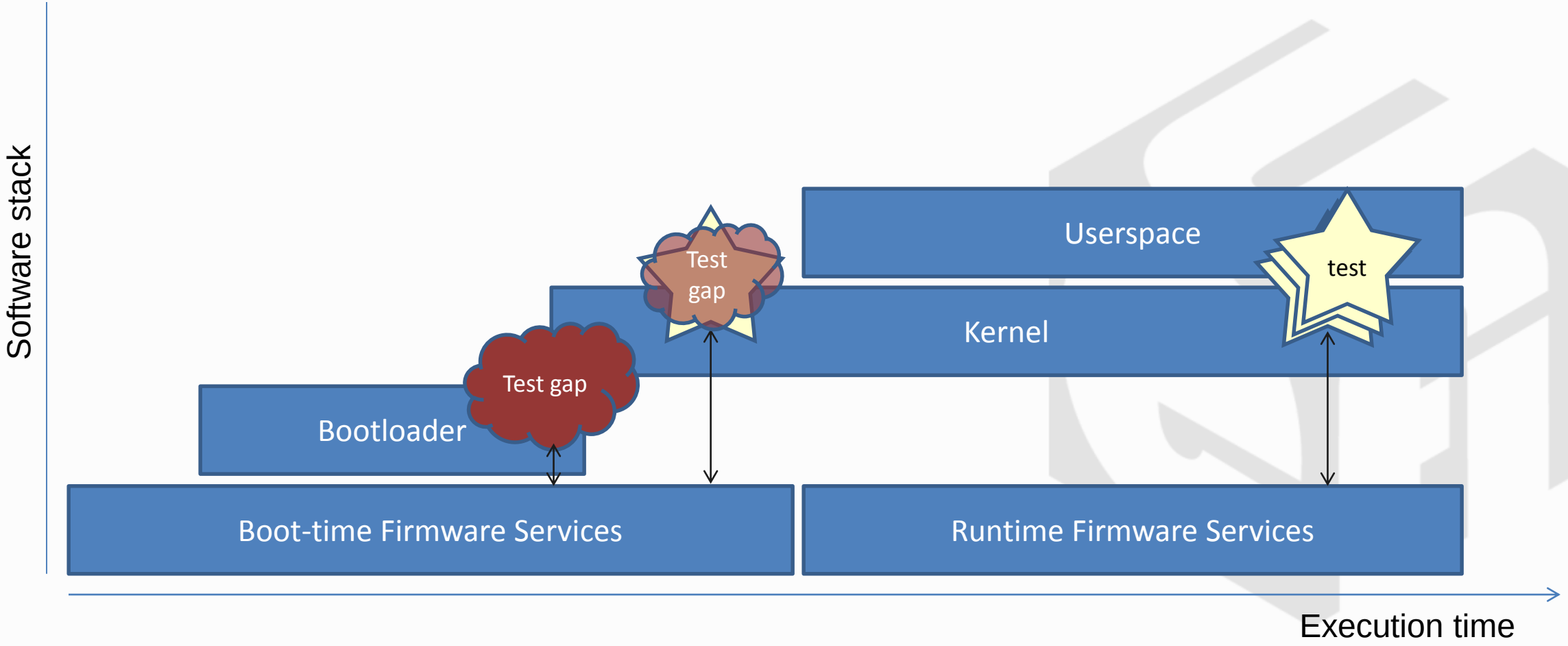
What is it not?

- A new test suite
- A replacement for existing test suites
- A certification tool for Linux or UEFI compliance
- A proprietary Intel tool

The Details


Linux UEFI validation operating system
Userspace
Test Manager
Test suite 1
Test suite 2
...
Test suite n-1
Kernel
Test suite n
Bootloader
Firmware
*Future scope

Covers Entire Execution Cycle



Future Development

Tests for bootloaders

Automation

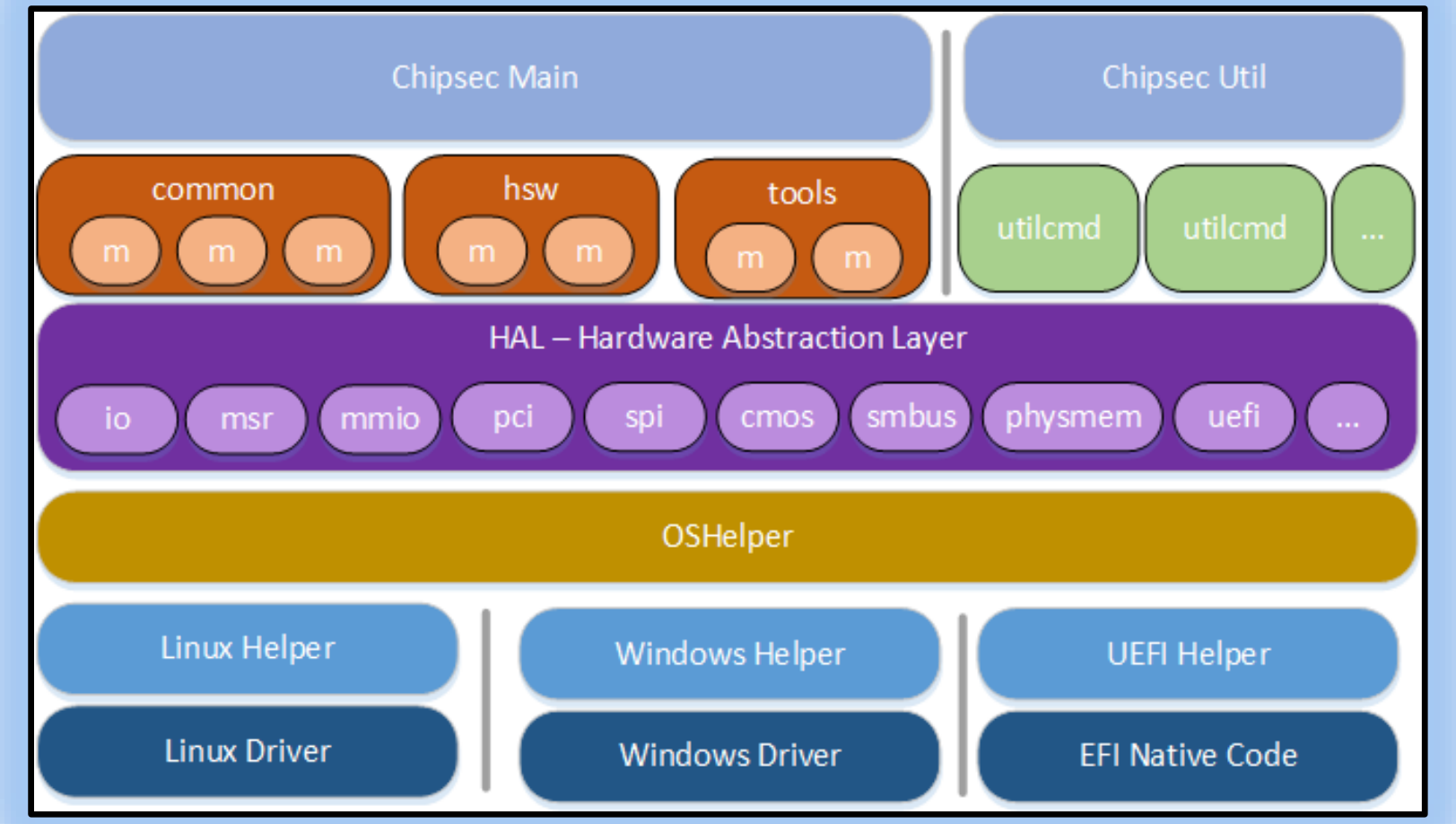
UEFI Capsule tests

UEFI Network stack tests

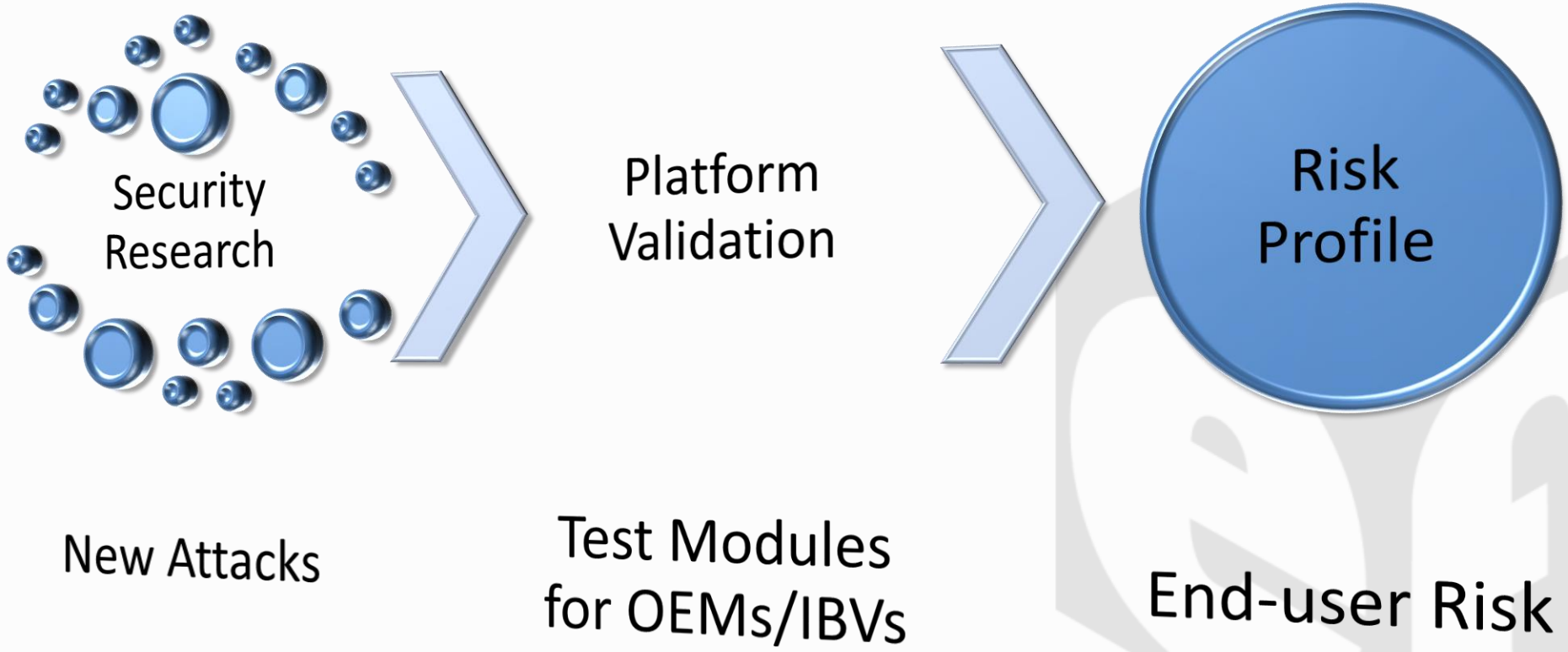
UEFI Secure Boot tests

CHIPSEC - Platform Security Assessment Framework

A single test designed to run in multiple environments



How do we raise the bar?



Empowering End-Users to Make a Risk Decision

Known Threats and CHIPSEC modules

Issue	CHIPSEC Module	Public Details
SMRAM Locking	common.smm	CanSecWest 2006
BIOS Keyboard Buffer Sanitization	common.bios_kbrd_buffer	DEFCON 16 2008
SMRR Configuration	common.smrr	ITL 2009 CanSecWest 2009
BIOS Protection	common.bios_wp	BlackHat USA 2009 CanSecWest 2013 Black Hat 2013 NoSuchCon 2013 Flashrom
SPI Controller Locking	common.spi_lock	Flashrom Copernicus
BIOS Interface Locking	common.bios_ts	PoC 2007
Access Control for Secure Boot Keys	common.secureboot.keys	UEFI 2.4 Spec
Access Control for Secure Boot Variables	common.secureboot.variables	UEFI 2.4 Spec

Example: BIOS Write Protection

Black Hat USA 2013 "BIOS Security" – MITRE (Kovah, Butterworth, Kallenberg)
NoSuchCon 2013 "BIOS Chronomancy: Fixing the Static Root of Trust for Measurement" – MITRE (Kovah, Butterworth, Kallenberg)

Is BIOS correctly protected?

▶ `common.bios_wp`

```
[+] imported chipsec.modules.common.bios_wp
[x][ ================================================================
[x][ Module: BIOS Region Write Protection
[x][ ================================================================
BIOS Control (BDF 0:31:0 + 0xDC) = 0x2A
[05]    SMM_BWP = 1 (SMM BIOS Write Protection)
[04]    TSS     = 0 (Top Swap Status)
[01]    BLE     = 1 (BIOS Lock Enable)
[00]    BIOSWE  = 0 (BIOS Write Enable)

[+] BIOS region write protection is enabled (writes restricted to SMM)

[*] BIOS Region: Base = 0x00500000, Limit = 0x00FFFFFF
SPI Protected Ranges
------------------------------------------------------------
PRx (offset) | Value    | Base     | Limit    | WP? | RP?
------------------------------------------------------------
PR0 (74)     | 00000000 | 00000000 | 00000000 | 0   | 0
PR1 (78)     | 8FFF0F40 | 00F40000 | 00FFF000 | 1   | 0
PR2 (7C)     | 8EDF0EB1 | 00EB1000 | 00EDF000 | 1   | 0
PR3 (80)     | 8EB00EB0 | 00EB0000 | 00EB0000 | 1   | 0
PR4 (84)     | 8EAF0C00 | 00C00000 | 00EAF000 | 1   | 0

[!] SPI protected ranges write-protect parts of BIOS region (other parts of BIOS can be modified)

[+] PASSED: BIOS is write protected
```

Direct HW Access for Manual Testing

Examples:

```
chipsec_util msr 0x200
chipsec_util mem 0x0 0x41E 0x20
chipsec_util pci enumerate
chipsec_util pci 0x0 0x1F 0x0 0xDC byte
chipsec_util io 0x61 byte
chipsec_util mmcfg 0 0x1F 0 0xDC 1 0x1
chipsec_util cmos dump
chipsec_util ucode id
chipsec_util smi 0x01 0xFF
chipsec_util idt 0
chipsec_util cpuid 1
chipsec_util spi read 0x700000 0x100000 bios.bin
chipsec_util decode spi.bin
chipsec_util uefi var-list
..
```

Forensics

Live system firmware analysis

```
chipsec_util spi info
chipsec_util spi dump rom.bin
chipsec_util spi read 0x700000 0x100000 bios.bin
chipsec_util uefi var-list
chipsec_util uefi var-read db D719B2CB-3D3A-4596-A3BC-DAD00E67656F db.bin
```

Offline system firmware analysis

```
chipsec_util uefi keys PK.bin
chipsec_util uefi nvram vss bios.bin
chipsec_util uefi decode rom.bin
chipsec_util decode rom.bin
```

Moving Forward

Test tools complement the SCT, but **the community can do more!**

Changing our development philosophy?

- "Testing shows the presence, not the absence of bugs" (*Dijkstra,1970*)
- Better Living Through Tools? (*Zimmer, 2013*)

Getting code coverage closer to 100%?

- Internal Intel effort using DDT with EDK II
- Moving to KLEE (open source)

"Infrastructure for automatic code checking" (coreboot)

- Automated system including KLEE, Splint, Frama-C

What Now?

Linux UEFI Validation Operating System

Download the code…
https://github.com/01org/luv-yocto

Download test images …
https://01.org/linux-uefi-validation

Submit your contributions

Questions?

- luv@lists.01.org
- ricardo.neri@intel.com
- matt.fleming@intel.com

CHIPSEC

Get the tool
https://github.com/chipsec/chipsec

Submit your contributions

Questions?

- chipsec@intel.com
- vincent.zimmer@intel.com

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

presented by