

presented by



UEFI Test Tools For Linux Developers

Brian Richardson – Intel Corporation
Alex Hung – Canonical, Ltd.

Agenda



- UEFI & Linux Interoperability
- Using FWTS with UEFI
- Using CHIPSEC with UEFI
- Using LuvOS with UEFI
- Next Steps

Agenda



- UEFI & Linux Interoperability
- Using FWTS with UEFI
- Using CHIPSEC with UEFI
- Using LuvOS with UEFI
- Next Steps

UEFI & Linux Interoperability



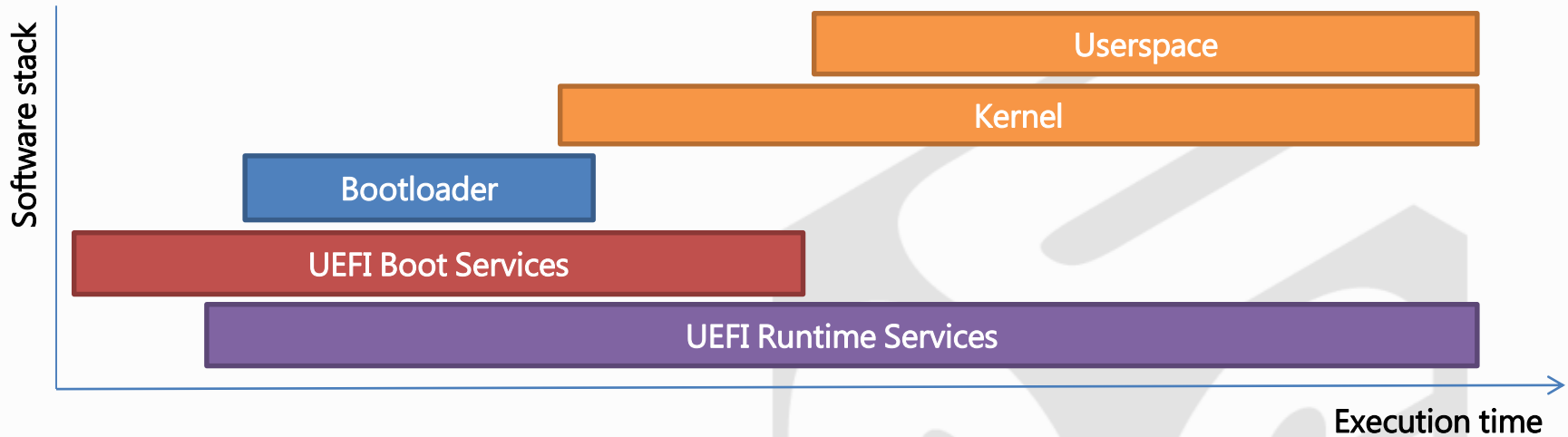
Improvements can be made in UEFI/Linux testing

- Look for UEFI problems prior to distro releases
- Tests are too focused on UEFI boot services
- Improve coverage for UEFI runtime services

Quality of firmware implementations varies

- Can open up new attack surfaces
- Example: "Samsung bricking"
<http://mjg59.dreamwidth.org/22855.html>

UEFI & Linux Software Stack



Used by bootloader and pre-OS applications (test, provision, etc.)

Large surface, terminated at boot

Traditionally tested using the UEFI Self-Certification Test (SCT)

Used by OS after boot services are terminated by bootloader

Small surface, resident at runtime

Traditionally tested/exercised by OS vendor validation plan

UEFI Runtime Needs More Attention



Many UEFI features are available at OS runtime

- Boot Order (**efibootmgr**)
- Capsule (system & peripheral firmware)
- Certificate Revocation (Secure Boot dbx)

Runtime variable access can have major issues

- Samsung bricking, Capsule buffer overflow, ...

UEFI Runtime Services Need to be Tested with Linux

Agenda



- UEFI & Linux Interoperability
- **Using FWTS with UEFI**
- Using CHIPSEC with UEFI
- Using LuvOS with UEFI
- Next Steps

Firmware Test Suite



What is Firmware Test Suite (FWTS)?

- Open-source Linux tool that automates firmware checking
- Detect bugs and advise firmware engineers
 - Test interactions between Linux & firmware
 - Gather firmware data for debug

What Does FWTS Test?

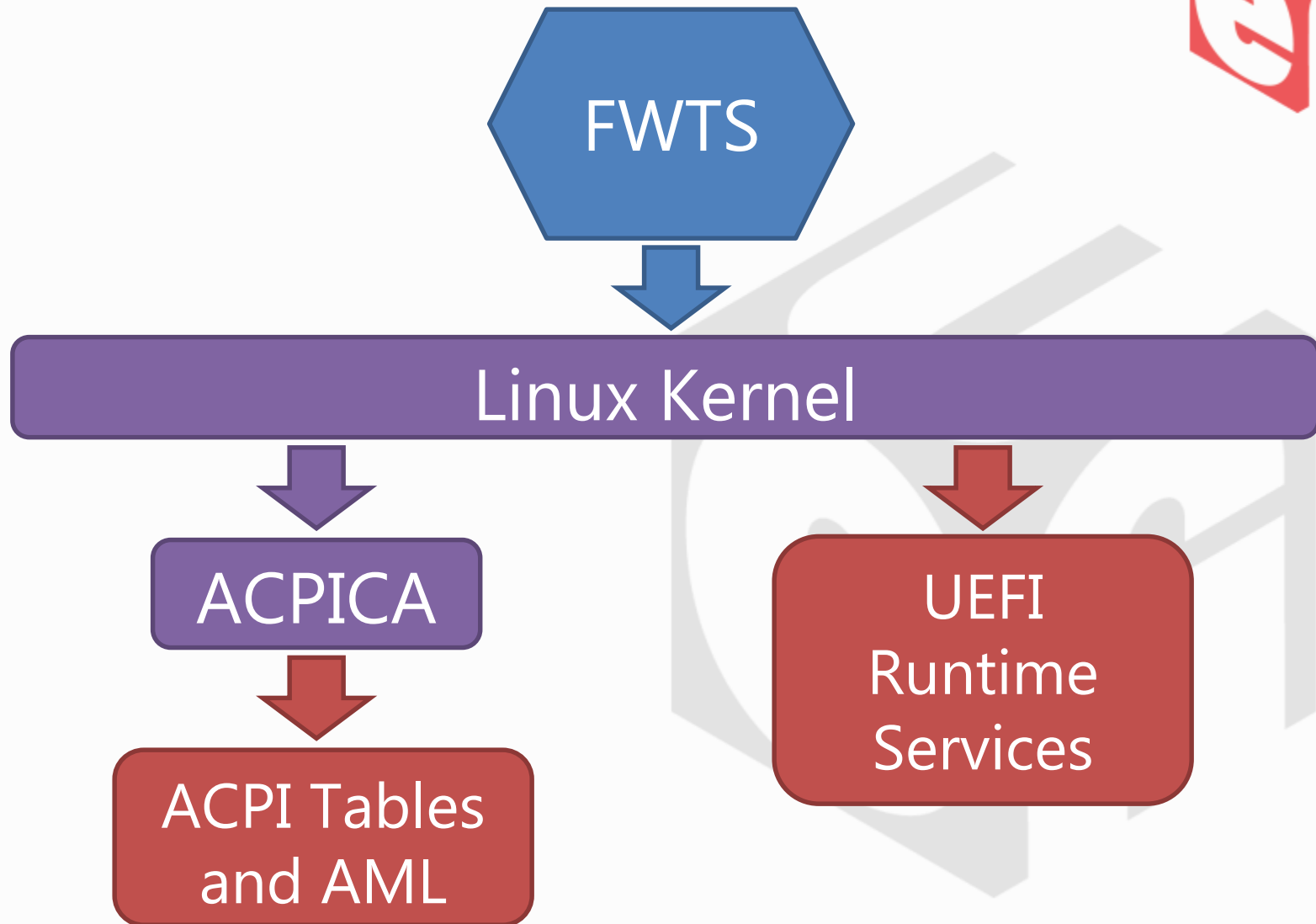


- Firmware Standards: ACPI, SMBIOS, UEFI, ...
- Hardware Standards: PCI/PCIe, x86 CPU
- System Level: Sleep and hibernate, interrupt configuration, and kernel log
- ... and many others

A complete list is available @

<https://wiki.ubuntu.com/Kernel/Reference/fwts>

FWTS Architecture



UEFI Test Items



Tests	Descriptions
securebootcert	UEFI secure boot
uefirtmisc	UEFI miscellaneous runtime service interface
uefirttime	UEFI Runtime service time interface
uefirtvariable	UEFI Runtime service variable interface

ACPI Test Items



Tests	Descriptions
acpitables	Check ACPI table settings sanity
checksum	Check ACPI table checksum
fadt	Check FADT SCI_EN enabled
method	Check ACPI DSDT Method Semantic
dmar/mcfg etc.	Verifies corresponding ACPI tables
Others...	Fan, C-/P- states etc.

Firmware Test Suite Live



- FWTS-Live – Bootable USB image
- Boot and run FWTS w/o installation



FWTS: Live In Action



```
Firmware Test Suite

                          Select Tests
This will run a suite of firmware tests that will check the BIOS
and ACPI tables. It can also find issues that can cause Linux
problems.

The default below is to run just all the Batch Tests, but you can
select more tests below if required.

Please select below (using cursor up/down and space) and press
enter to continue:

(*) 1 All Batch Tests
( ) 2 Select Individual Tests
( ) 3 Abort Testing

< OK >      <Cancel>      < Help >
```

FWTS: Live In Action (Cont'd)



```
Firmware Test Suite
-----
Running Batch Tests
So far: 7 passed, 0 failed, 0 warnings, 3 aborted, 0 skipped, 1 info only
Test ACPI Wakealarm.
Running test #8: Multiple wakealarm firing tests.

[Progress bar showing 22% completion]
```

FWTS: Live In Action (Cont'd)



```
Firmware Test Suite

Testing Complete
The results can be found on the USB stick in the
the directory: /fwts/11102011/1327/results.log

Do you want to view the results log now?

< Yes >      < No >
```


FWTS Result Log



```
alexhung@earth: ~  
Command: "fwts uefirttime".  
Running tests: uefirttime.  
  
uefirttime: UEFI Runtime service time interface tests.  
-----  
Test 1 of 4: Test UEFI RT service get time interface.  
PASSED: Test 1, UEFI runtime service GetTime interface test passed.  
  
Test 2 of 4: Test UEFI RT service set time interface.  
PASSED: Test 2, UEFI runtime service SetTime interface test passed.  
  
Test 3 of 4: Test UEFI RT service get wakeup time interface.  
FAILED [HIGH] UEFIRuntimeTimeFieldBadYear: Test 3, Time returned an invalid year  
0, should be between 1900 and 9999.  
  
Test 4 of 4: Test UEFI RT service set wakeup time interface.  
FAILED [HIGH] UEFIRuntimeSetWakeupTimeVerify: Test 4, Failed to verify wakeup  
time after change.  
  
=====
```

2 passed, 2 failed, 0 warning, 0 aborted, 0 skipped, 0 info only.

```
=====
```

31,0-1 25%

FWTS Result Log (Cont'd)



```
alexhung@earth: ~
=====
Critical failures: NONE

High failures: 2
 uefirttime: Time returned an invalid year 0, should be between 1900 and 9999.
 uefirttime: Failed to verify wakeup time after change.

Medium failures: NONE

Low failures: NONE

Other failures: NONE

Test          |Pass |Fail |Abort|Warn |Skip |Info |
-----+-----+-----+-----+-----+-----+
uefirttime    |  2|  2|   0|   0|   0|   0|
-----+-----+-----+-----+-----+
Total:        |  2|  2|   0|   0|   0|   0|
-----+-----+-----+-----+-----+
55,1 Bot
```

Agenda



- UEFI & Linux Interoperability
- Using FWTS with UEFI
- **Using CHIPSEC with UEFI**
- Using LuvOS with UEFI
- Next Steps

Testing Platform Security (CHIPSEC)

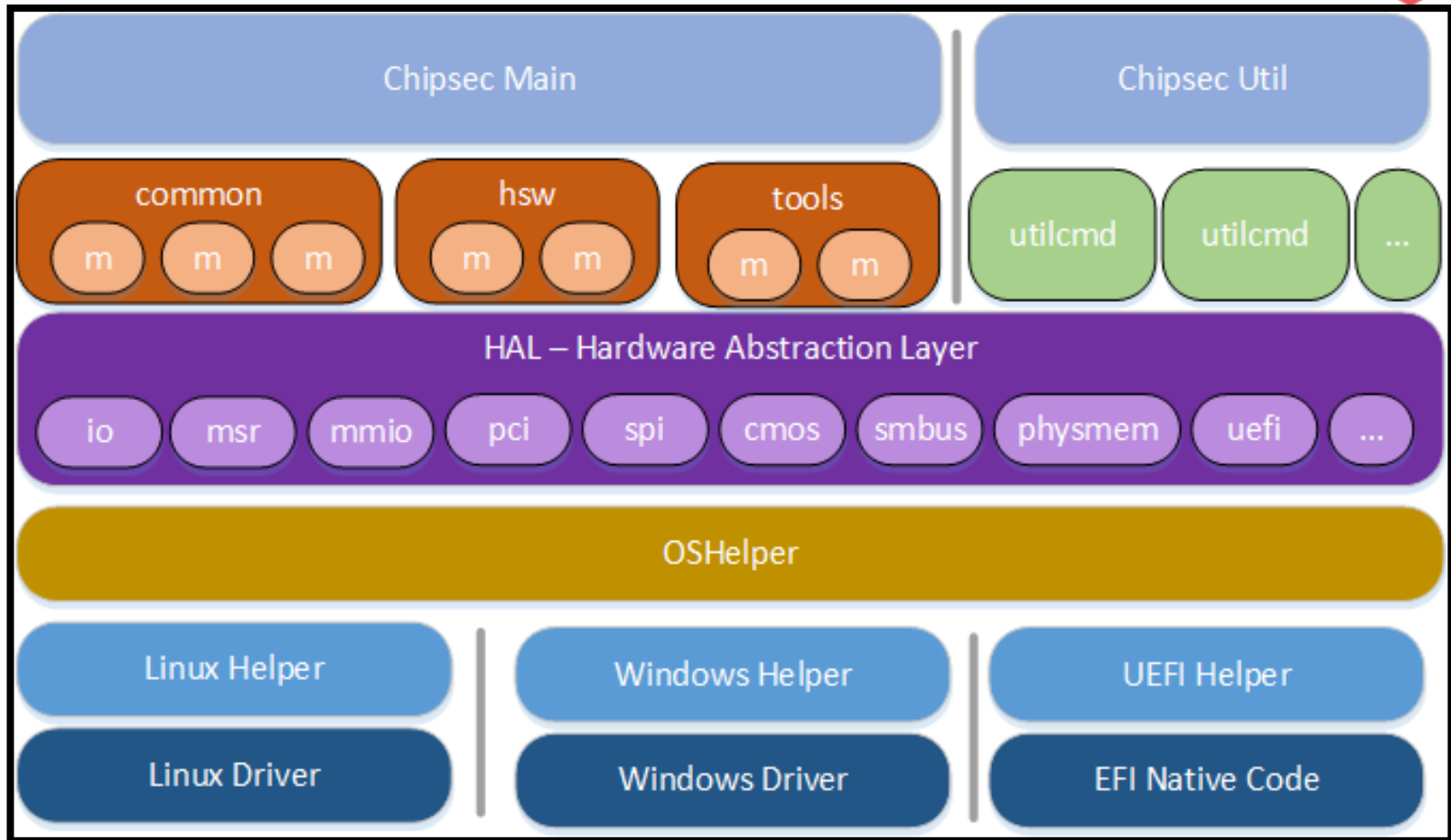


Platform Security Assessment Framework
<https://github.com/chipsec/chipsec>

Single test suite for Linux, Windows & UEFI

- Analyze hardware, firmware & components
- Extensible security test suite (Python)
- Security assessment tools for low level components/interfaces
- Forensic capabilities for firmware

CHIPSEC: Platform Security Assessment Framework



Examples Of Test Modules



Direct H/W Access for Manual Testing

```
chipsec_util msr 0x200  
chipsec_util smi 0x01 0xFF
```

Forensics – Live firmware analysis

```
chipsec_util spi dump rom.bin  
chipsec_util spi read 0x700000 0x100000 bios.bin  
chipsec_util uefi var-list
```

Forensics – Offline firmware analysis

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

Ex: BIOS Write Protection



```
[+] 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)
```

Based on MITRE at *Black Hat USA 2013* and *NoSuchCon 2013*

Is BIOS correctly protected?

```
[+] 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
```

Testing Using Known Threats



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

Leverage Community Knowledge for Firmware Testing

Agenda



- UEFI & Linux Interoperability
- Using FWTS with UEFI
- Using CHIPSEC with UEFI
- **Using LuvOS with UEFI**
- Next Steps

The logo features a white square with a red keyhole icon, positioned above a red square with a white keyhole icon. The word "Linux" is written in white above the word "UEFI Validation", which is in a larger white font.

Linux UEFI Validation

01.org/linux-uefi-validation



- Common manager for existing UEFI test suites
- Focused on testing Linux with UEFI firmware
- Test new components prior to use in new distros
- Removes focus on UEFI pre-boot environment
- Distributed under MIT license



01.org/linux-uefi-validation

- ❌ New test suite
- ❌ Proprietary Intel tool
- ❌ Replacement for existing test suites
- ❌ Certification tool for Linux/UEFI compliance

Hands-off Testing

1. Boot the USB image
2. LuvOS collects results.
3. There is no Step 3.

```
[ - ] efiwarfs
[+] test_create... passed
[+] test_create_empty... passed
[+] test_create_read... passed
[+] test_delete... passed
[+] test_zero_size_delete... passed
[+] test_open_unlink... passed
[+] test_valid_filenames... passed
[+] test_invalid_filenames... passed
[ - ] fwts
[+] bios_info... passed
[+] version... passed
[+] acpiinfo... passed
[+] mtrr... 25 failures!
[+] klog... 2 failures!
[+] oops... passed
[+] acpitables... passed
[+] apicinstance... passed
[+] autobrightness... passed
[+] checksum... passed
[+] cstates... passed
[+] dmar... passed
[+] fadt... passed
[+] fan... passed
[+] mcfg... 1 failures!
[+] method... 29 failures!
[+] osilinux... skipped
[+] pcc... passed
[+] syntaxcheck... 122 failures!
[+] wakealarm... passed
[+] wmi... passed
[+] apicedge... passed
```

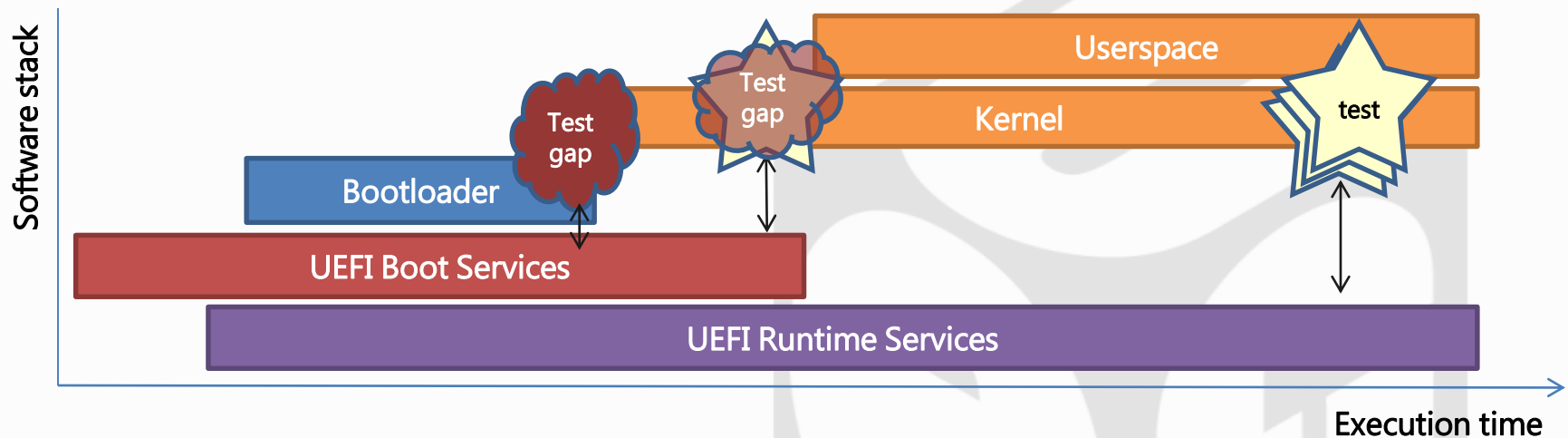
LuvOS Features



- Included test suites: FWTS, CHIPSEC, BITS & efivarfs
- Future UEFI tests: Capsule, Network Stack, Secure Boot and Linux bootloaders

A screenshot of the GitHub repository page for '01org / luv-yocto'. The page shows the repository name, a search bar, and navigation links like 'Explore', 'Features', 'Enterprise', and 'Blog'. It also displays statistics such as '26,697 commits', '7 branches', '2 releases', and '185 contributors'. A commit history table is visible, showing a recent commit by 'mfleming' titled 'Revert "core-image-efi-initramfs: Add chipsec test"'. The right sidebar contains links for 'Code', 'Issues', 'Pull Requests', 'Pulse', and 'Graphs'.

LuvOS... Covers Entire Execution Cycle



Use LuvOS to Bridge Gaps in Linux/UEFI Validation

Agenda



- UEFI & Linux Interoperability
- Using FWTS with UEFI
- Using CHIPSEC with UEFI
- Using LuvOS with UEFI
- **Next Steps**

Next Steps



Download and run the tools ...

- FWTS - <https://odm.ubuntu.com/>
- CHIPSEC - <https://github.com/chipsec/chipsec>
- LuvOS - <https://01.org/linux-uefi-validation>

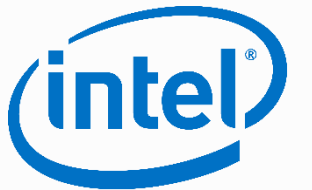
Questions? Contributions? Bugs?!?

- fwts-announce@lists.ubuntu.com
- chipsec@intel.com
- <https://lists.01.org/mailman/listinfo/luv>



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

presented by



Look Inside.™

The Canonical logo, a dark purple rectangle with the word "CANONICAL" in white, uppercase, sans-serif font, with a small trademark symbol (TM) to the right.

CANONICAL™