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

- **Software stack**
  - Bootloader
  - UEFI Boot Services
  - UEFI Runtime Services
  - Kernel
  - Userspace

- **Execution time**
  - 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

LinuxCon 2014
www.uefi.org
UEFI Runtime Needs More Attention

Many UEFI features are available at OS runtime

• Boot Order (\texttt{efibootmgr})
• Capsule (system & peripheral firmware)
• Certificate Revocation (Secure Boot dbx)

Runtime variable access can have major issues

• Samsung bricking, Capsule buffer overflow, ...
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

- **FWTS**
  - **Linux Kernel**
    - **ACPICA**
      - **ACPI Tables and AML**
    - **UEFI Runtime Services**
# UEFI Test Items

<table>
<thead>
<tr>
<th>Tests</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>securebootcert</td>
<td>UEFI <strong>secure boot</strong></td>
</tr>
<tr>
<td>uefirtmisc</td>
<td>UEFI <strong>miscellaneous</strong> runtime service interface</td>
</tr>
<tr>
<td>uefirttime</td>
<td>UEFI Runtime service <strong>time</strong> interface</td>
</tr>
<tr>
<td>uefirtvariable</td>
<td>UEFI Runtime service <strong>variable</strong> interface</td>
</tr>
</tbody>
</table>
## ACPI Test Items

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

ACPI Test Items

LinuxCon 2014 | www.uefi.org
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
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.

22%
FWTS: Live In Action (Cont’d)

Firmware Test Suite

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

Do you want to view the results log now?

< Yes >  < No >
FWTS Result Log

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.
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

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

Forensics – Live firmware analysis

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

Forensics – Offline firmware analysis

```bash
chipsec_util uefi keys PK.bin
chipsec_util uefi nvram vss bios.bin
chipsec_util uefi decode rom.bin
```
**Ex: BIOS Write Protection**

```plaintext
[+] imported chipsec.modules.common.bios_wp
[x] Module: BIOS Region Write Protection

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

<table>
<thead>
<tr>
<th>SPI Protected Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRx (offset)</td>
</tr>
<tr>
<td>----------------------</td>
</tr>
<tr>
<td>PR0 (74)</td>
</tr>
<tr>
<td>PR1 (78)</td>
</tr>
<tr>
<td>PR2 (7C)</td>
</tr>
<tr>
<td>PR3 (80)</td>
</tr>
<tr>
<td>PR4 (84)</td>
</tr>
</tbody>
</table>

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

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

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

Is BIOS correctly protected?
## Testing Using Known Threats

<table>
<thead>
<tr>
<th>Issue</th>
<th>CHIPSEC Module</th>
<th>Public Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMRAM Locking</td>
<td>common.smm</td>
<td>CanSecWest 2006</td>
</tr>
<tr>
<td>BIOS Keyboard Buffer Sanitization</td>
<td>common.bios_kbrd_buffer</td>
<td>DEFCON 16 2008</td>
</tr>
<tr>
<td>SMRR Configuration</td>
<td>common.smrr</td>
<td>ITL 2009</td>
</tr>
<tr>
<td>BIOS Protection</td>
<td>common.bios_wp</td>
<td>BlackHat USA 2009</td>
</tr>
<tr>
<td>BIOS Interface Locking</td>
<td>common.bios_ts</td>
<td>CanSecWest 2013</td>
</tr>
<tr>
<td>SPI Controller Locking</td>
<td>common.spi_lock</td>
<td>Black Hat 2013</td>
</tr>
<tr>
<td>BIOS Interface Locking</td>
<td>common.bios_ts</td>
<td>NoSuchCon 2013</td>
</tr>
<tr>
<td>Access Control for Secure Boot Keys</td>
<td>common.secureboot.keys</td>
<td>Flashrom</td>
</tr>
<tr>
<td>Access Control for Secure Boot Variables</td>
<td>common.secureboot.variables</td>
<td></td>
</tr>
</tbody>
</table>

**Leverage Community Knowledge for Firmware Testing**
Agenda

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

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