Rust based Shim-Firmware for Confidential Container

Jiewen Yao, Principal Engineer, Intel
February, 2023
Notices & Disclaimers

• Intel Corporation (“Intel”) provides these materials as-is, with no express or implied warranties.

• No license (express, implied, by estoppel, or otherwise) to any intellectual-property rights is granted by this document.

• © Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands might be claimed as the property of others.
**Speaker**

**Jiewen Yao** is a principal engineer in the Intel Software and Advanced Technology Group. He has been engaged as a firmware developer for about 20 years. He is a member of the UEFI Security sub team, the TCG PC Client working group, and chairing DMTF SPDM Code Task Force.

He is the architect for Intel® TDX virtual firmware.
Agenda

- Background
- Why shim-firmware?
- TD-Shim Internal
Need of virtual firmware
## Virtual Firmware Solution

<table>
<thead>
<tr>
<th>Main Feature</th>
<th>SeaBIOS</th>
<th>OVMF</th>
<th>cloud-hypervisor-firmware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypervisor</td>
<td>XEN, KVM, ...</td>
<td>XEN, KVM, ...</td>
<td>cloud-hypervisor, ...</td>
</tr>
<tr>
<td>Arch</td>
<td>16 bit</td>
<td>32bit/64bit</td>
<td>64bit</td>
</tr>
<tr>
<td>VMM-BIOS Entrypoint</td>
<td>16bit Reset Vector</td>
<td>16bit Reset Vector</td>
<td>ELF Entrypoint</td>
</tr>
<tr>
<td>BIOS-OS Interface</td>
<td>Legacy 16bit INT</td>
<td>UEFI Specification</td>
<td>Linux Boot Protocol</td>
</tr>
</tbody>
</table>

**Gap Analysis (TDX)**
1. Entrypoint – 32bit Reset Vector
2. MP Wakeup – special wakeup structure
3. Memory Initialize – memory accept before use
4. DMA Management – shared/private memory switch
5. Measurement – Runtime Measurement Register (RTMR) extend

**Solution**
- N/A
- TDVF (upstreamed)
- TD-SHIM
td-shim

- A **lightweight** virtual firmware for **confidential container** environment.
- Written in Rust
- Support Intel ® TDX
  - [https://github.com/confidential-containers/td-shim](https://github.com/confidential-containers/td-shim)

- **Responsibility**
  - Own the 1\textsuperscript{st} instruction (reset vector) of a TD
  - Provide the required boot information (memory map, CPU info) to the next phase (payload)
  - Build the chain-of-trust from Intel ® TDX-module to the next phase
td-shim boot vs. TDVF boot

TD-Payload (OS, Bare-metal)
- Memory Map (E820), ACPI Table (MADT)
- TD HOB (Memory Info)

TD-Shim

VMM

OS Loader (GRUB)
- UEFI Memory Map, UEFI Runtime ACPI Table (MADT, DSDT,...)

OS Kernel (Kernel, Initrd)

TDVF (OVMF)
- TD HOB (Memory Info)

VMM
## td-shim vs. TDVF

<table>
<thead>
<tr>
<th></th>
<th>TD Shim</th>
<th>TDVF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use Case</strong></td>
<td>Confidential Container, Small Service TD</td>
<td>Confidential VM, Rich Service TD</td>
</tr>
<tr>
<td><strong>Language</strong></td>
<td>RUST (no-std) + ASM</td>
<td>C + ASM</td>
</tr>
<tr>
<td><strong>UEFI Service &amp; Features</strong></td>
<td>NO</td>
<td>Network, File System, etc</td>
</tr>
<tr>
<td><strong>OS Runtime</strong></td>
<td>NO</td>
<td>UEFI RT, ACPI ASL</td>
</tr>
<tr>
<td><strong>Device Driver</strong></td>
<td>NO</td>
<td>Virtio, PCI, etc</td>
</tr>
<tr>
<td><strong>ACPI Table (MP Support)</strong></td>
<td>Static table only (MADT,..). No DSDT.</td>
<td>All (MADT, DSDT, ...)</td>
</tr>
<tr>
<td><strong>IRQ Info</strong></td>
<td>Other (Boot Param, ...)</td>
<td>ACPI DSDT</td>
</tr>
<tr>
<td><strong>Memory Map</strong></td>
<td>E820 table</td>
<td>UEFI Memory Map</td>
</tr>
<tr>
<td><strong>Trusted Boot</strong></td>
<td>YES (RTMR + EventLog)</td>
<td>YES (RTMR + EventLog)</td>
</tr>
<tr>
<td><strong>Secure Boot</strong></td>
<td>Optional</td>
<td>Optional (UEFI Secure Boot)</td>
</tr>
<tr>
<td><strong>Image Size (release)</strong></td>
<td>140K (w/o SecureBoot)</td>
<td>4M by default.</td>
</tr>
<tr>
<td></td>
<td>270K (full feature, w/ SecureBoot)</td>
<td></td>
</tr>
</tbody>
</table>
td-shim boot

- Parse TD hob
- Measure TD hob
- Get memory info
- Accept Memory
- Locate/Load payload
- Jump to payload

- Park AP
- Switch to long mode
- Setup stack
- Jump to shim/main()

Confidential Container

- TD-Payload

- TD-Shim
  - Reset Vector

VMM
td-shim use cases

Direct Kernel Boot
- Linux Image (vmlinuz, bzimage)
- Linux boot protocol
- TD-Shim
- Reset Vector

Confidential Container

Migration TD
- Migration TD Core
- Executable payload
- TD-Shim
- Reset Vector

Service TD
td-shim feature

• Trusted Boot
  • https://github.com/confidential-containers/td-shim/blob/main/doc/tdshim_spec.md#guideline

• Secure Boot
  • https://github.com/confidential-containers/td-shim/blob/main/doc/secure_boot.md
Trusted Boot

- td-shim extends measurement to TD runtime measurement register (RTMR)
- td-shim provides event log (CCEL) to reproduce the value in RTMR.
- Attestation can be based upon MR register or event log.
Trusted Boot

Confidential Computing Env

- TD Payload
- TD-Shim

VMM

TD-Config (TD HOB)

Measurement Registers

- RTMR3
- RTMR2
- RTMR1
- RTMR0

Initial Boot Block

Firmware

OS

App

Special
Secure Boot

- Verify the next component before launch
- Need to provision the known good public key and secure version number (SVN)
- Payload attestation can be based upon SVN value, not image hash.
Secure Boot

Confidential Computing Env
- TD Payload
  - SVN
  - public key signature
- TD-Shim
  - key hash
  - min svn

TD-Config (TD HOB)

VMM

Measurement Registers
- RTMR3
- RTMR2
- RTMR1
- RTMR0

TDMR

Firmware
- Initial Boot Block

App
- OS

Special
Other Features

• Data Execution Protection (DEP)
  • Page table based enforcement.
  • DataPage = Non-Executable
  • CodePage = Read-Only

• Control Flow Enforcement (Intel® CET)
  • Backward-Edge control flow - Shadow Stack (SS)
  • Forward-Edge control flow – Indirect Branch Tracking (IBT)
    • Depend upon compiler (TBD)
Tools

• tee_info_hash tool

• Payload reference calculator

• metadata checker
Test

- fuzzing-test: afl-fuzz, cargo-fuzz
- static code scan: cargo-clippy, rudra, Prusti, MIRAI
- vulnerable crate scan: cargo-deny
- general test:
  - unit test coverage: https://github.com/confidential-containers/td-shim/blob/main/doc/unit_test_coverage.md
Reference

• Intel® TDX

• Virtual Firmware for Intel® Trust Domain Extensions
  • https://cfp.osfc.io/osfc2020/talk/CRKZB8/

• Enabling Rust for UEFI firmware
  • https://cfp.osfc.io/osfc2020/talk/SLFJTN/