Firmware Update in Total Compute Platform

Manish Badarkhe and Davidson Kumaresan
22/09/2022

© 2022 Arm
Agenda

- Total compute software stack
- Firmware update spec revision
- EFI UpdateCapsule runtime service
- Call flow
- Work yet to be done
TC Software Stack
- Setup and Discovery interfaces
- Scheduling interfaces
- Messaging interfaces
- Memory management interfaces
- Notification interfaces
- Status reporting interfaces
PSA Firmware Update terminology

- Banks
- Image Directory
- Staging area
- States
Firmware - FIP

- FIP in TC has
  - BL2 (loader)
  - SCP_RAMFW
  - BL31 (secure monitor)
  - BL32 (Hafnium)
  - Secure Partitions – Trusted services and trusted OS (trusty/optee)
  - BL33 (U-Boot)

- This FIP is referred as firmware here and the images present in the FIP are upgradable.
Partition Layout

FIP as a partition inside GPT image

- Protective MBR
- Primary GPT Header
- Entries 5-128
- FIP_A partition
- Remaining partitions
- Secondary GPT Header
- Entries 5-128

Legend:
- Protected partitions
- Reserved partitions
- Backup reserved partitions
- Available partitions

GPT - GUID partition table
LBA - Logical Block Addressing
## Metadata

### Table 5: Metadata version 1

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset (bytes)</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crc_32</td>
<td>0h</td>
<td>4h</td>
<td></td>
</tr>
<tr>
<td>version</td>
<td>4h</td>
<td>4h</td>
<td></td>
</tr>
<tr>
<td>active_index</td>
<td>8h</td>
<td>4h</td>
<td></td>
</tr>
<tr>
<td>previous_active_index</td>
<td>1Ch</td>
<td>4h</td>
<td></td>
</tr>
<tr>
<td>img_entry [#images]</td>
<td>10h</td>
<td>#images.(20h + #banks.18h)</td>
<td>array of aggregate in Table 6</td>
</tr>
</tbody>
</table>

### Table 6: Metadata image entry version 1

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset (bytes)</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>img_type_uid</td>
<td>0h</td>
<td>10h</td>
<td>UUID identifying the image type</td>
</tr>
<tr>
<td>location_uid</td>
<td>10h</td>
<td>10h</td>
<td>the UUID of the storage volume where the image is located</td>
</tr>
<tr>
<td>img_bank_info[#banks]</td>
<td>20h</td>
<td>18h.#banks</td>
<td>the properties of images with img_type_uid in the different FW banks</td>
</tr>
</tbody>
</table>

### Table 7: Image properties in a given FW bank version 1

<table>
<thead>
<tr>
<th>Field</th>
<th>Offset (bytes)</th>
<th>Size (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>img_uuid</td>
<td>0h</td>
<td>10h</td>
<td>the uuid of the image in this bank</td>
</tr>
</tbody>
</table>
| accepted| 10h            | 4h           | • [0]: bit describing the image acceptance status – 1 means the image is accepted  
|         |                |              | • [31:1]: MBZ                                                              |
| reserved| 14h            | 4h           | reserved (MBZ)                                                             |
TC software stack for Firmware Update
U-Boot

- It is the update client
- Use the UpdateCapsule runtime service of EFI
 EFI System Resource Table (ESRT)

SoC system containing n firmware resources

Each of the firmware resources maps to an entry in the ESRT Firmware Resource Entry Array. (Note that other data associated with each entry is not shown for simplicity. Refer to ESRT implementation documentation for details.)
Firmware Management Protocol APIs

- GetImageInfo
- GetImage
- SetImage
- CheckImage
- GetPackageInfo
- SetPackageInfo
Capsules

- To do capsule update, we have to convert the FIP image into a capsule
- To create capsule, GenerateCapsule from the edk2 project has to be used

```
edk2/BaseTools/BinWrappers/PosixLike/GenerateCapsule -e -o efi_capsule --fw-version 1 --lsv 0 --guid 0d5c011f-0776-5b38-8e81-36fbd6743e2 --verbose --update-image-index 0 --verbose fip-tc.bin
```
EFI UpdateCapsule

Process UEFI Capsule

UEFI Capsule

- UEFI Capsule Header
- FMP Header
- Auth Info
- Payload Header (Extensible)
- Payload

SetImage()

Authenticate

System Firmware

- FMP Driver
  - ImageType Id
  - GUID A
  - Public Key(s)

Flash

ESRT Table

- GUID A

Publish

Update

FMP = UEFI Firmware Management Protocol
GUID = Globally Unique Identifier

www.uefi.org
Firmware Update Secure Partition

- It is implemented in the TrustedServices project which is maintained in the [https://git.trustedfirmware.org/](https://git.trustedfirmware.org/)
- It will run at S-EL0 as a secure partition
- It is the implementation of the arm PSA firmware update specification. It implements all the PSA firmware update APIs.
- NOR flash driver required to read and write to the NOR flash device.
- GPT partition driver required to parse the GPT partition header and to get the partition information from the flash.
Firmware Update APIs

- Discover
- BeginStaging
- EndStaging
- CancelStaging
- Open
- WriteStream
- ReadStream
- Commit
- AcceptImage
- SelectPrevious
Communication

<table>
<thead>
<tr>
<th>EFI UpdateCapsule service</th>
<th>U-Boot</th>
<th>Firmware Update SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>FMP_GetImageInfo</td>
<td>FWU_Open ← FWU_Open</td>
<td>FWU_Open</td>
</tr>
<tr>
<td></td>
<td>FWU_Read ← FWU_Read</td>
<td>FWU_Read</td>
</tr>
<tr>
<td></td>
<td>FWU_BeginStaging ← FWU_BeginStaging</td>
<td>FWU_BeginStaging</td>
</tr>
<tr>
<td></td>
<td>FWU_Open ← FWU_Open</td>
<td>FWU_Open</td>
</tr>
<tr>
<td>FMP_SetImage</td>
<td>FWU_Write ← FWU_Write</td>
<td>FWU_Write</td>
</tr>
<tr>
<td></td>
<td>FWU_Commit ← FWU_Commit</td>
<td>FWU_Commit</td>
</tr>
<tr>
<td></td>
<td>FWU_EndStaging ← FWU_EndStaging</td>
<td>FWU_EndStaging</td>
</tr>
</tbody>
</table>
Communication

- U-Boot places the message into the shared memory and sends a `ffa_direct_msg_req()` to the Firmware Update secure partition.
- Firmware Update secure partition receives the `ffa_direct_msg_req()` and reads the content from the shared memory.
- Firmware Update secure partition performs the action mentioned in the message and puts the reply message in the shared buffer and sends the `ffa_direct_msg_resp()`.
Communication
Call Flow – During Boot

- **U-Boot**
  - Register FMP driver
  - FWU_Discover
    - Information about the FWU SP

- **FWU SP**
  - FMP_GetImageInfo
    - FWU_Open (Image directory)
    - fd of image directory
  - Populate ESRT
    - FWU_Read (fd of Image directory)
    - contents of Image directory
Call flow – During update

1. Create a staging area
2. Get the FD for the firmware to be updated.

Transfer the firmware image from U-Boot to FWU SP

1. Set state = trial
2. Update active_index and prev_active_index

Create a capsule

Place it in memory

Execute U-Boot command line app

UpdateCapsule service invoked

Find the FMP driver that handles the Image UUID

Call SetImage of that FMP driver

Update ESRT
Boot Flow 1/2 – BL2 Execution

1. Load FWU metadata to global metadata structure
2. Do sanity check on FWU metadata
3. CRC32 validation on FWU metadata
4. CRC or sanity check failures, repeat step 1 to 3 on backup metadata, if that still fails the checks, then panic.

Set I/O policy for fwu metadata image source

Set I/O policies for all the images loaded by BL2 using active firmware bank index
Boot Flow 2/2 – Trial run detection

- `bl2_main`
- `bl2_image_load_v2`
- `bl_common`
- `auth_mod`
- `fwu`

**BL2 common code**
- `bl2_load_images()`: load all images as per image_info array
- `load_auth_image()`: load and authenticate image

**Drivers**
- `auth_mod_verify_img0`
- `auth_nvctr()`
- `fwu_is_trial_run_state()`: Get whether system running in regular or trial state to decide on platform NV counter update
Features to be implemented

- Actual runtime update via OTA
- UEFI capsule authentication from UEFI spec
- PSA Image authentication from PSA spec
- Anti rollback counter from PSA spec
- No acceptance tests for the firmware - the newly updated firmware is implicitly accepted - from PSA spec
- Selecting previous image/bank from PSA spec
- Recovery mode from PSA spec
Reference

+ ARM PSA firmware update spec
  https://developer.arm.com/documentation/den0118/a/?lang=en

  • EFI runtime services – Sec 8
  • UpdateCapsule – Sec 8.5.3
  • Firmware Management Protocol – Sec 23
  • ESRT – Sec 23.3
  • GPT partition layout – Sec 5

+ Trusted Services Project - https://git.trustedfirmware.org/TS/trusted-services.git/

+ Source code and patches will be part of the next Total Compute release.
Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
ありがとう
Asante
Merci
감사합니다
धन्यवाद
धन्यवाद
شكرًا
תודה