SCMI server in secure World

Etienne Carriere
Souvik Chakravarty
Vincent Guittot
Agenda

● Introduction to SCMI
● Why SCMI server in secure world?
● SCMI server in OP-TEE
● SCMI server in other secure environments
● SCP firmware
● Accessing SCMI Server from Linux
● Current Status
● Limitation & Constraints
● Armv7-A Specificities
● Next
Introduction to SCMI

- SCMI - System Control and Management Interface

- Provides standardized interface for power management between power co-processor/system-controller and its clients in a SoC.
  - Examples of clients: Linux (Normal World), TEE or TF-A (Secure World).
  - Each client can have its own access permissions to power resources.

- Protocols Supported:
  - Base, Power Domain, System Power, Performance Domain, Clock, Sensor, Reset Domain.

- More Details:
  
Why SCMI server in secure world?

- Some SoCs may not have:
  - a dedicated power co-processor/system-controller, or
  - capability to support multiple clients, e.g., they might support only 1 secure channel.

- In such systems the Secure World:
  - has access to all Power Resources.
  - exposes non-secure Power Resources to Normal World through SCMI.
  - provides Power Co-Processor/System-Controller services to Normal World through SCMI Services.
SCMI server in OP-TEE

- Use OP-TEE thread context as entry point
  - Related to mailbox and entry in OP-TEE in yield (non fastcall) mode

- Parse OP-TEE message to catch SCMI messages
  - Keep same message structure as HW mailbox

- Implementation based on SCP-firmware source tree as reference
  - Decided to have optee_os.git fetch/build SCP-firmware.git

- Integrate SCMI server in privileged core, not userland: driver layer
SCMI server in other secure environnements

- OP-TEE is the first secure environment considered

- But there are cases for using other secure contexts
  - In TF-A
  - In TF-M
  - Bare metal in a dedicated secure partition for Arm v8.4

Therefore an interest in sharing a common code based for SCMI server implementation:

⇒ SCP-firmware.git is the current SCMI reference implementation
SCP firmware

- Repo: [https://github.com/ARM-software/SCP-firmware](https://github.com/ARM-software/SCP-firmware)
  - Current dev available here: [SCMI server in optee](#) (study on PD, clock, perfs, SPCI Mbox)
  - And: [SCMI server in stm32mp1](#) (study on clock, reset, armv7-A & OP-TEE integ.)

- Current goal: build source tree from OP-TEE core source build
  - OP-TEE SCMI server support shall be native in SCP-firmware repository

- Changes needed
  - Adapt for OP-TEE: logs, memory alloc, virtual addressing, speculative exec.
  - Create OP-TEE platforms specific configuration and HAL drivers
  - Make modules working without multithreading
  - Disable notification mechanism in modules
  - Create Mailbox modules for call based notification (not interrupt based)
  - Framework entry to request processing of a pending message
Accessing SCMI server from Linux

- SCMI client available since v4.16
- Resources populated in PM framework
- Support multiple mailboxes and channels
  - Support only HW mailbox
    - Adding SMC mailbox
    - Adding SPCI
- Support multiple transport layer
Current Status

Development and prototypes done on FVP, Qemu, stm32mp1.

- FVP (armv8-A): prototyped SCMI clock, power domain, perfs with a SPCI mailbox.
- Qemu (armv7-A/armv8-A): SCMI clocks with a SPCI or SMC mailbox.
- Stm32mp1 (armv7-A): SCMI clocks and reset domain, on SPCI or SMC mailbox.

Upstream work in progress:

- Linux kernel: SCMI v2.0 support (among which Reset Domain) posted to LKML.
- Linux SMC mailbox [https://patchwork.kernel.org/patch/9858641](https://patchwork.kernel.org/patch/9858641) (NXP, patch v3 avail.)
- SCP-firmware.git: support for OP-TEE SCMI server, for Reset Domain.

Upstream work planned:

- SPCI support, once matured, planned by Arm and Linaro.
- optee_os.git: fetch SCP-firmware sources, configure and build SCMI server.
Limitation & constraints

- Mono request
  - Handle request sequentially

- Polling mode during SCMI services
  - Can’t take advantage of interruption of secure resources like for I2C transfer

- OP-TEE RPC service not fully supported

- No notifications to agents
  - SCMI notification support mandates a non-secure interrupt
  - Find a non-secure interrupt for secure to non-secure event signaling.

- SCP-firmware implementation regarding Cortex-A speculative execution

- Memory footprint when secure world is located in small secure RAM (<400kB)
Armv7-A Specificities

- Armv7-A OP-TEE core embeds its own secure monitor (OP-TEE $sm$) (where Armv8-A decouples secure monitor (BL31/EL3) from OP-TEE (BL32/EL1))

  SCMI impacts in secure monitor must be ported into OP-TEE $sm$ component
  - Using SPCI mailbox mandates SPCI implemented in OP-TEE $sm$
  - Using SMC mailbox, OP-TEE $sm$ can simply route the SMC into generic EL1

- Impacts of integration of SCMI server in OP-TEE $sm$
  - Possible memory footprint issue with OP-TEE pager ($CFG\_WITH\_PAGER=y$). $sm$ executes in OP-TEE “unpaged”: code must be resident in secure RAM.
Next steps

- Send pull request for current development
- Enable SMC mailbox support
- Support interrupt handling for drivers in SCMI server
- Enable notification to agent:
  - Need a non-secure interrupt raised from secure world
- Optimize memory footprint
  - SCMI with clock/reset currently estimated ~12kB ROM.
- Optimize performance
Thank you

Join Linaro to accelerate deployment of your Arm-based solutions through collaboration

contactus@linaro.org