TF-M SPM Backends & API Interfaces

Kevin Peng

March 2022

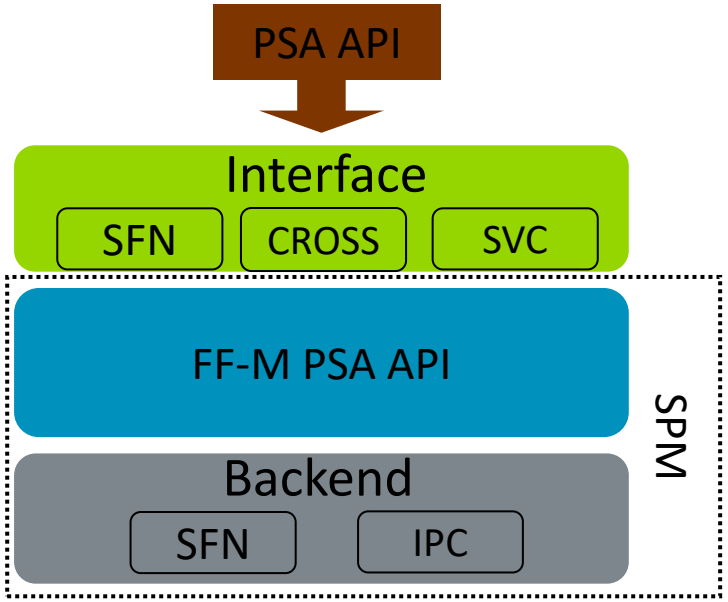
Background

- FF-M 1.1 extensions introduced the SFN Model
  - A new programming model of Secure Partitions
- To support this model of Secure Partitions
  - TF-M introduced the SFN backend (aka SFN Model SPM) to differentiate some implementations with the “IPC Model”
  - TF-M also introduced 3 types of PSA API interfaces
- Backends are about how the TF-M SPM works
- The PSA API interfaces are how Secure Partitions interact with the SPM
- Interface + Backend together decide how the whole SPE works

Backends Overview

- The SFN backend SPM:
  - No isolation within the SPE
  - Single-thread execution within the whole SPE – the NS Agent Partition thread
  - One single stack only
  - No context switch/scheduler
  - Supports SFN Model Partitions only
  - Targets for resource constraint devices
- The IPC backend SPM (aka the IPC Model)
  - Dedicated threads and stacks for Secure Partitions
  - Has context switch/scheduler
  - Supports all level of isolations
  - Supported IPC Partitions only previously and SFN Partitions are supported recently

Interfaces



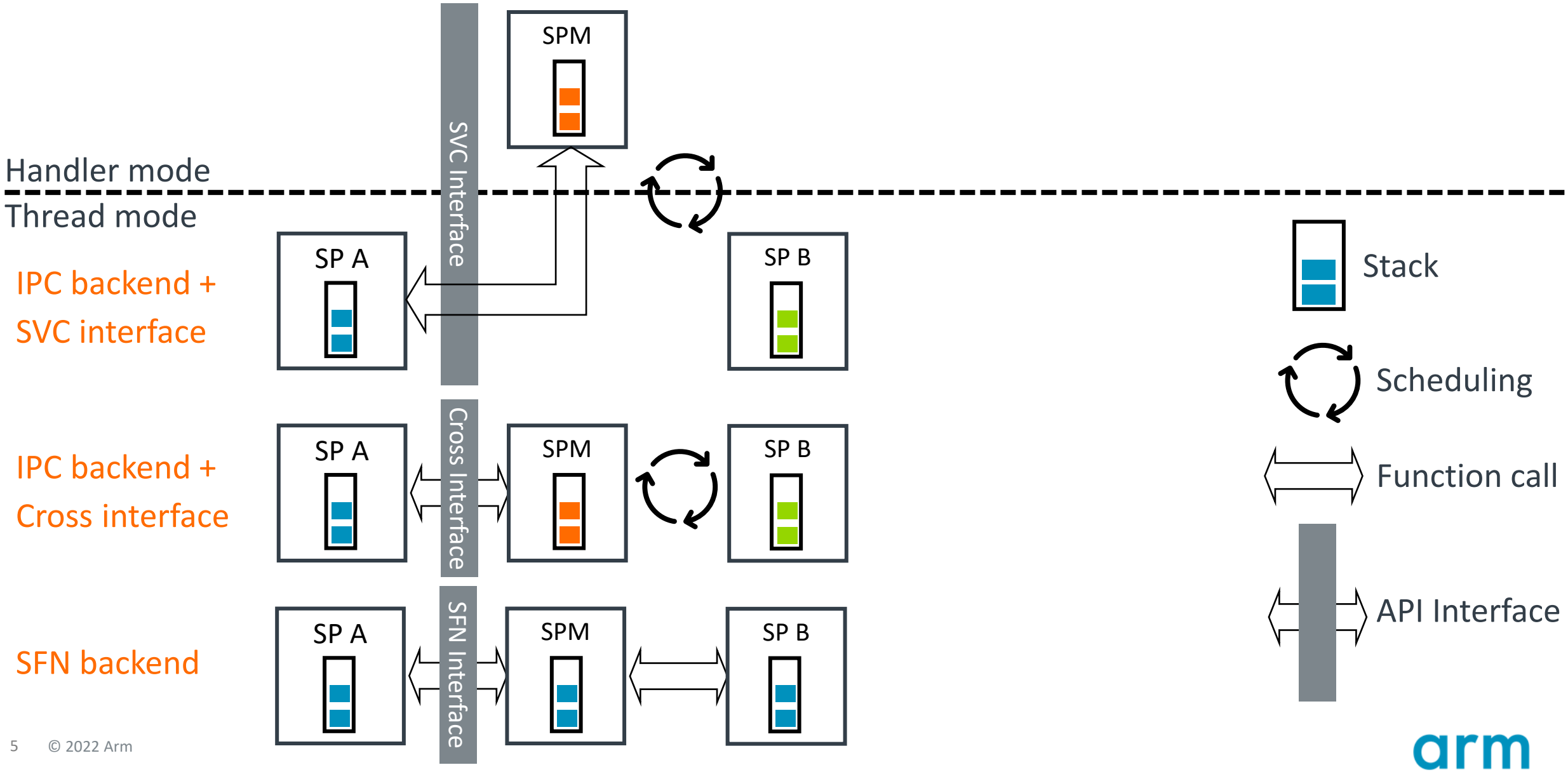
Interfaces used by backends

Backends	Isolation Levels		
	L1	L2	L3
SFN	SFN		
IPC	Cross	SVC	SVC

- SFN (psa_interface_sfn.c)
  - Designed for SFN Backend
  - Direct function call
- Cross call (psa_interface_cross.c)
  - Designed for IPC Backend + isolation L1
  - Does not change privilege
  - Switch to SPM stack and lock scheduler
  - SPM execution is Preemptable
- SVC (psa_interface_svc.c)
  - Designed for high isolation levels (L2 & L3)
  - SVC to Handler mode
  - Changes privileged mode
  - SPM execution is Non-preemptable

arm

Execution Models



Backend Operations

- comp_init_assuredly
  - Initializes the Secure Partition runtime structures
    - Major differences: signals, sync objects (IPC)
  - Initializes the Secure Partition threads
    - Major differences: SFN only have the NS agent thread
- system_run
  - Starts the system after initialization
    - IPC: updates boundaries and start scheduler
    - SFN: starts NS agent to initialize partitions
- messaging (for client APIs)
  - IPC: Sends messages (signals) to target SP, sets thread to runnable state and sets the current SP to block state
  - SFN: Function calls to the target Secure Function
- replying (called when SPs reply messages)
  - IPC: Wakes up the service requestor’s thread
  - SFN: Function return

Backend Operations – Cont'd

- wait – Secure Partitions wait for signals
  - IPC: Sets waiting signals and sets Partition thread to block state
  - SFN: Waits for signals with infinite loop
- wake_up – Secure Interrupts wakes up Partitions
  - IPC: Wake up the Secure Partition's thread
  - SFN: Nothing

SFN Model Secure Partition Support in IPC Backend

Treat SFN Partitions as if they were IPC Partitions

"Conceptually, for a single service named SERVICE1 in a Secure Partition manifest, the framework behaves *as if* it was the following IPC model entry point" – FF-M 1.1 extensions

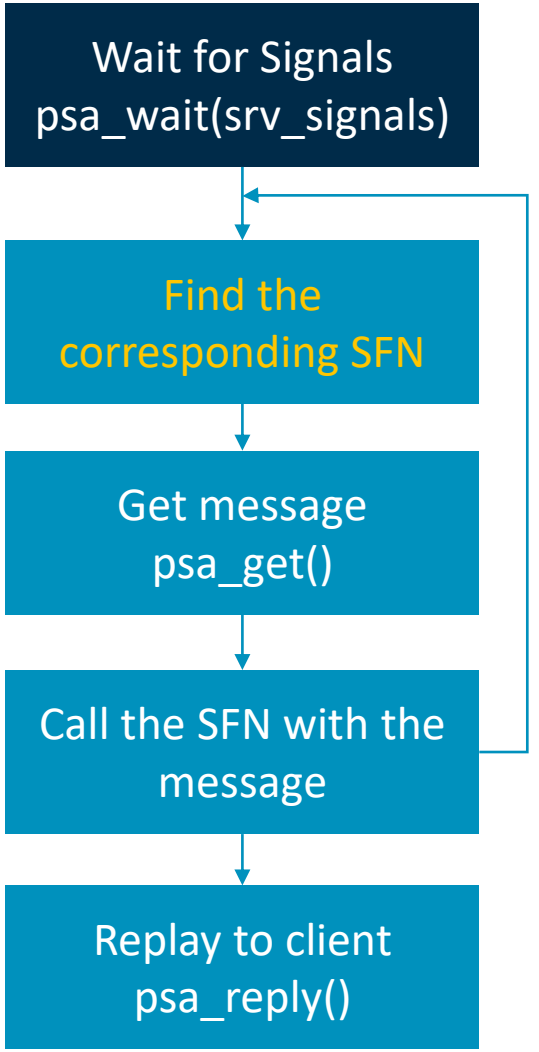
```
void sp_main(void)
{
    psa_msg_t msg;

    for (;;)
    {
        psa_wait(SERVICE1_SIGNAL, PSA_BLOCK);
        if (psa_get(SERVICE1_SIGNAL, &msg) == PSA_SUCCESS)
            psa_reply(msg.handle, service1_sfn(&msg));
    }
}
```

SFN Model Secure Partition Support in IPC Backend

Treat SFN Partitions as IPC Partitions

- Assigns signals for SFN Partitions
- Allocate threads for SFN Partitions
- Runs SFN Partitions in common thread codes
- All the above are agnostic to SFN Secure Partitions
- Leverages the existing IPC backend and interfaces which are mature



Summaries

Backends	Model of Secure Partitions		Isolation Levels			Interrupt Handling	
	IPC	SFN	L1	L2	L3	FLIH	SLIH
SFN		✓	SFN Interface			✓	✓
IPC	✓	✓	Cross interface	SVC Interface	SVC Interface	✓	✓

- The backend is selected by the *CONFIG_TFM_SPM_BACKEND* [*IPC, SFN*]
  - IPC backend is the default
- The interface is then selected by the build system according to isolation levels.

arm

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה