OpenIoT Summit Europe 2018

Compartmentalization in IoT

Trusted Firmware M Secure Partitioning

Miklos Balint
Ken Liu
Arm

Agenda

The right level of security

Hardware support

Compartmentalization scenarios

Interaction between isolated components

Challenges in IoT

High volume, low cost, low power

- Microcontrollers
  - Small die
  - No MMU (single, physical address space)
  - XIP Flash code
  - Small SRAM

Wide spectrum of use-cases

- Different threat models
- Scalable solutions

Holistic approach to IoT security needed

arm

Establishing the “right” level of security

Secure domain

Basic isolation – create a Secure Processing Environment

Protected TCB

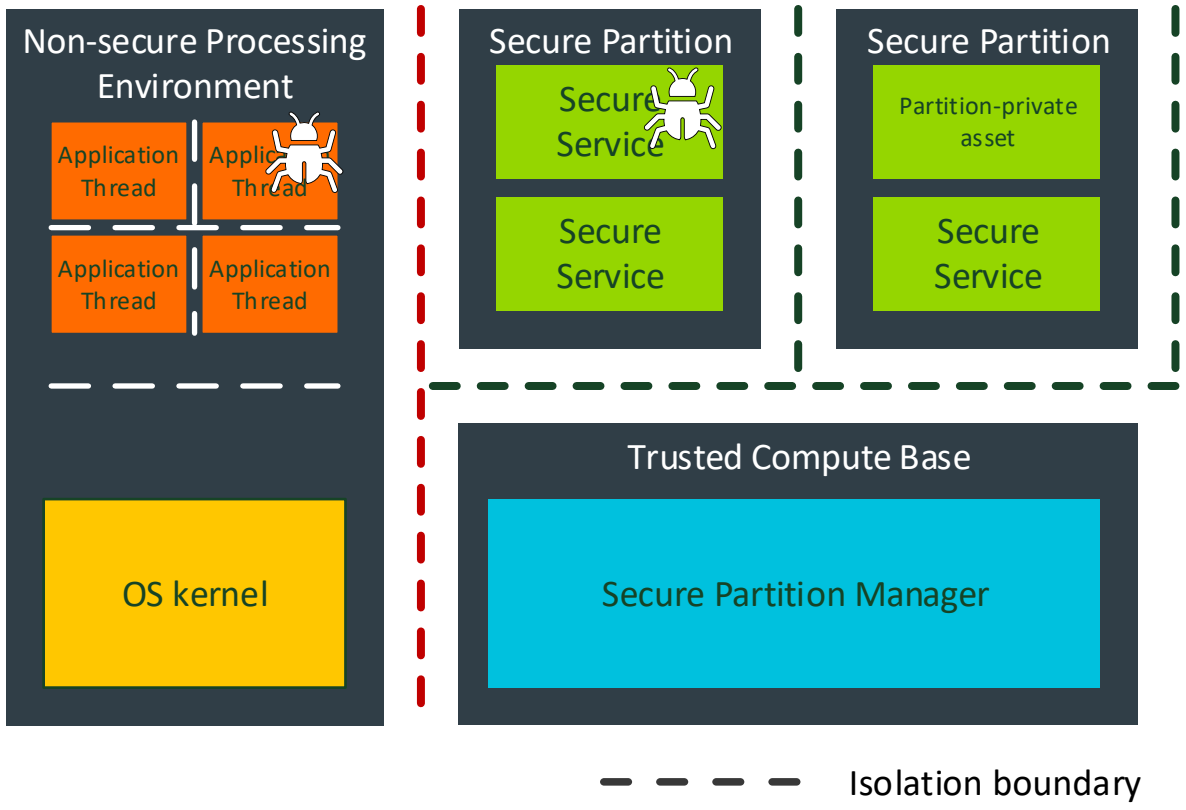
Separate Root of Trust from Secure Partitions within SPE

Multiple tenancy in secure PE

More robustness – isolate all partitions from each other

Non-Secure isolation

Access policies for NS threads



arm

Hardware isolation

… the foundation for software security

Physical isolation (e.g. dual-core system):

Dedicate cores/resources

Shared memory system or Mailbox

Concurrent execution

Temporal isolation (e.g. Arm-v8M):

Privilege control – using MPU

Secure/Non-secure states (Secure Attribution)

Shared Processing Element, resources

arm

Interaction scenarios

arm

Execution flows

Crossing boundaries in single processing element

Crossing from Non-secure to secure state

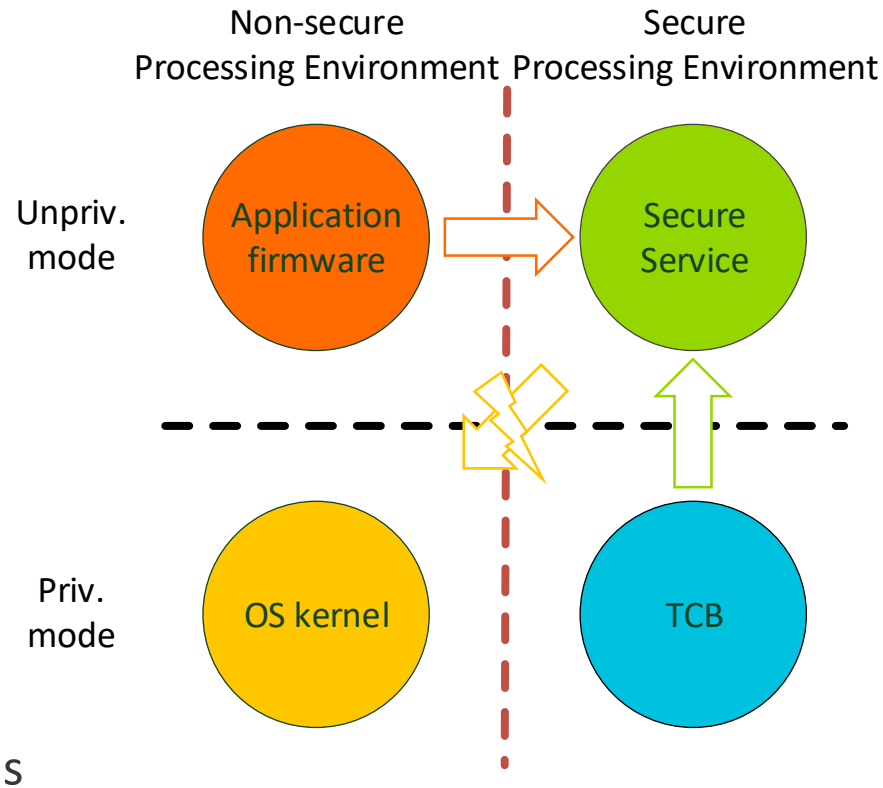
- Non-secure thread requests secure service

Isolated driver code

- ISR execution in unprivileged partition

Asynchronous events in non-secure PE

- Non-secure interrupt pre-empts secure operation
- Non-secure context awareness
- Concurrent secure service requests from non-secure threads



arm

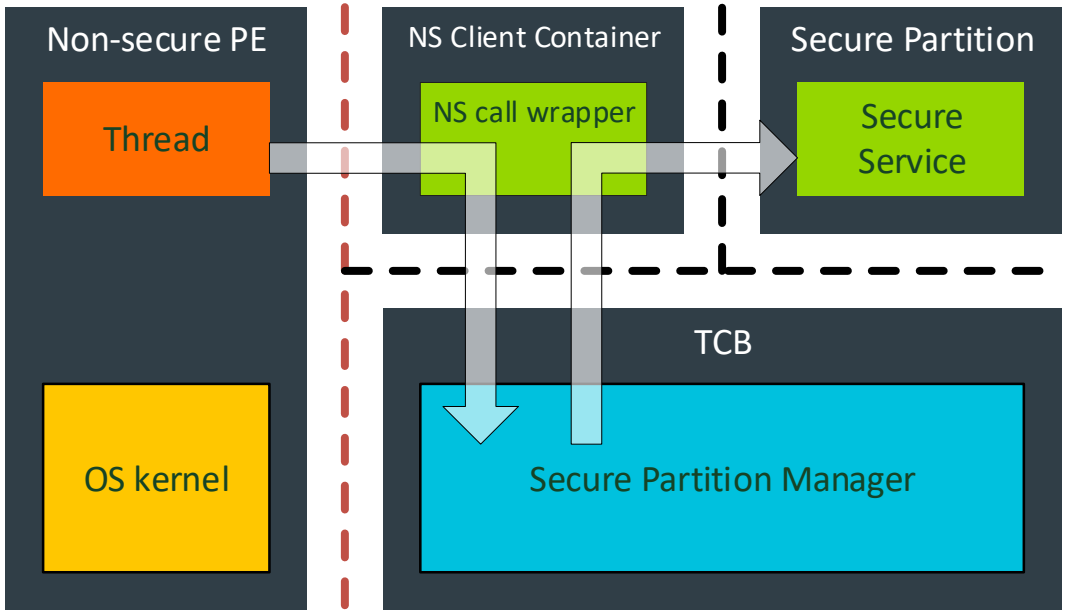
Non-secure call to secure service

Security state change only permitted using dedicated entry points

Wrapper function triggers privileged management code

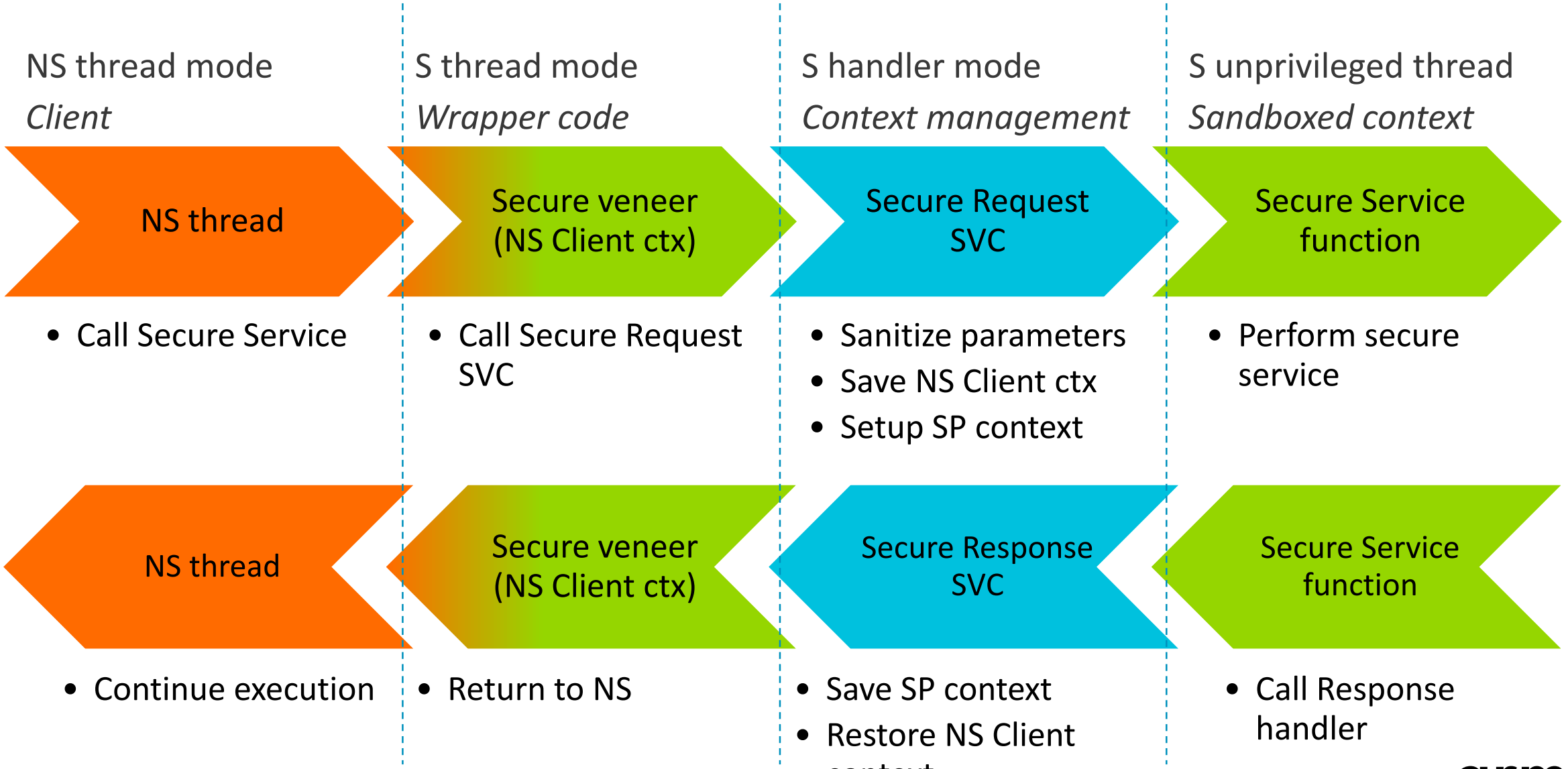
Secure Partition Management code

- Performs parameter sanitization
- Sets up Secure Partition (container)



arm

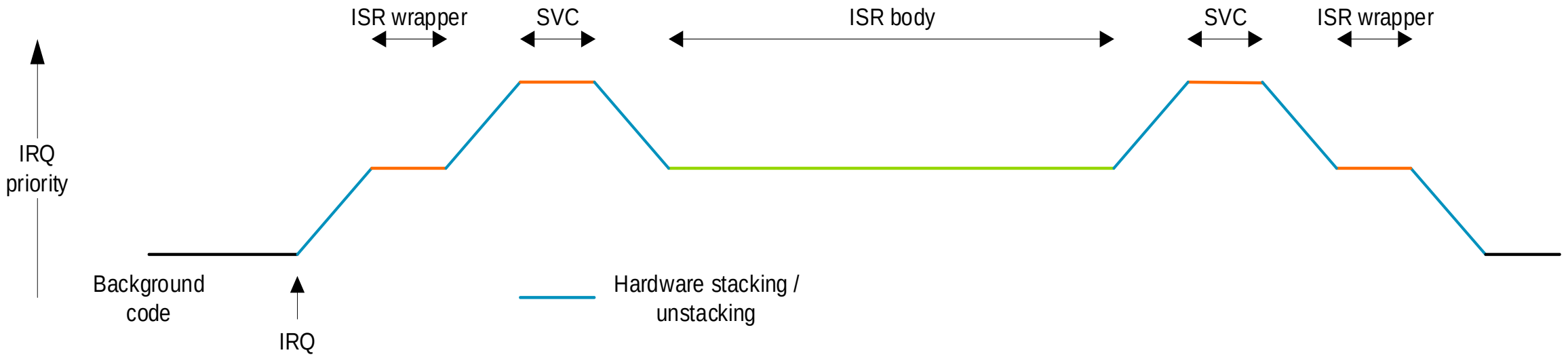
Non-secure call to secure service



arm

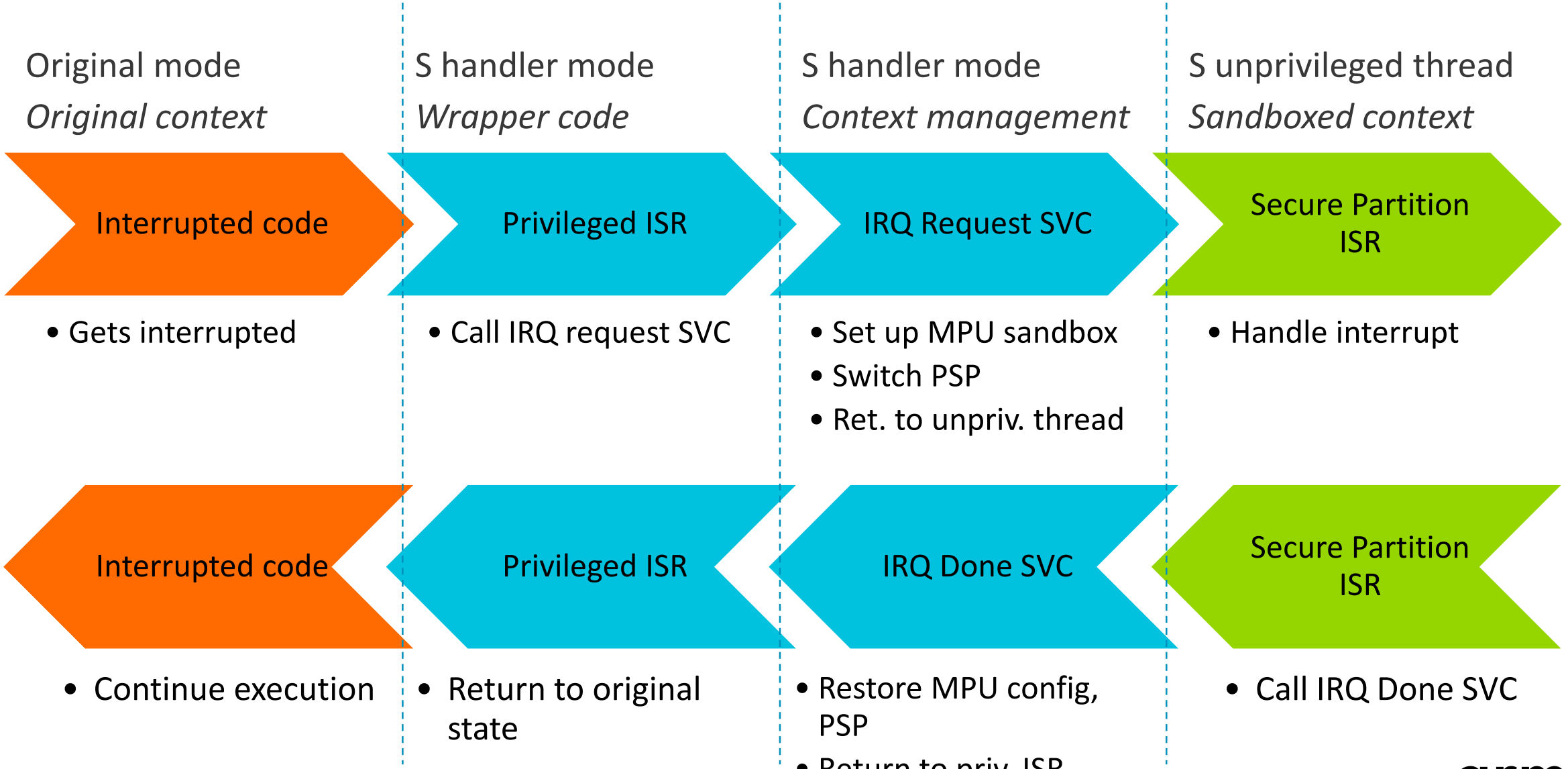
Secure interrupt deprivileging

- Privileged ISR is wrapper
  - Triggers Partition Manager
- Sandbox created
  - Returns to thread mode
- Secure Partition code
  - Executes deprivileged ISR



arm

Secure interrupt deprivileging



arm

Non-Secure interrupts

Pre-emption of secure execution

Non-secure thread is executing

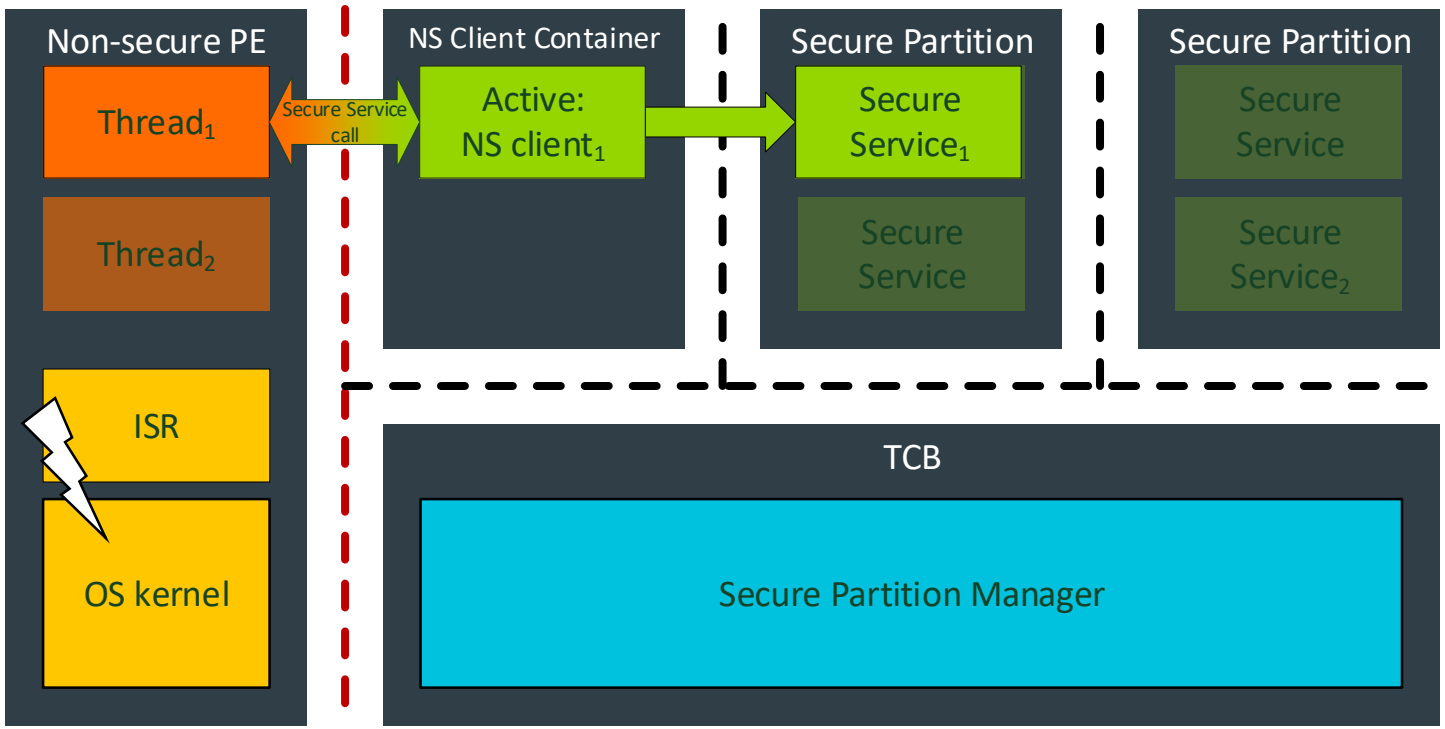
Thread calls Secure Service

Non-secure IRQ pre-empts operation

Secure context is stacked

Non-secure ISR is executed

Return from ISR resumes secure execution



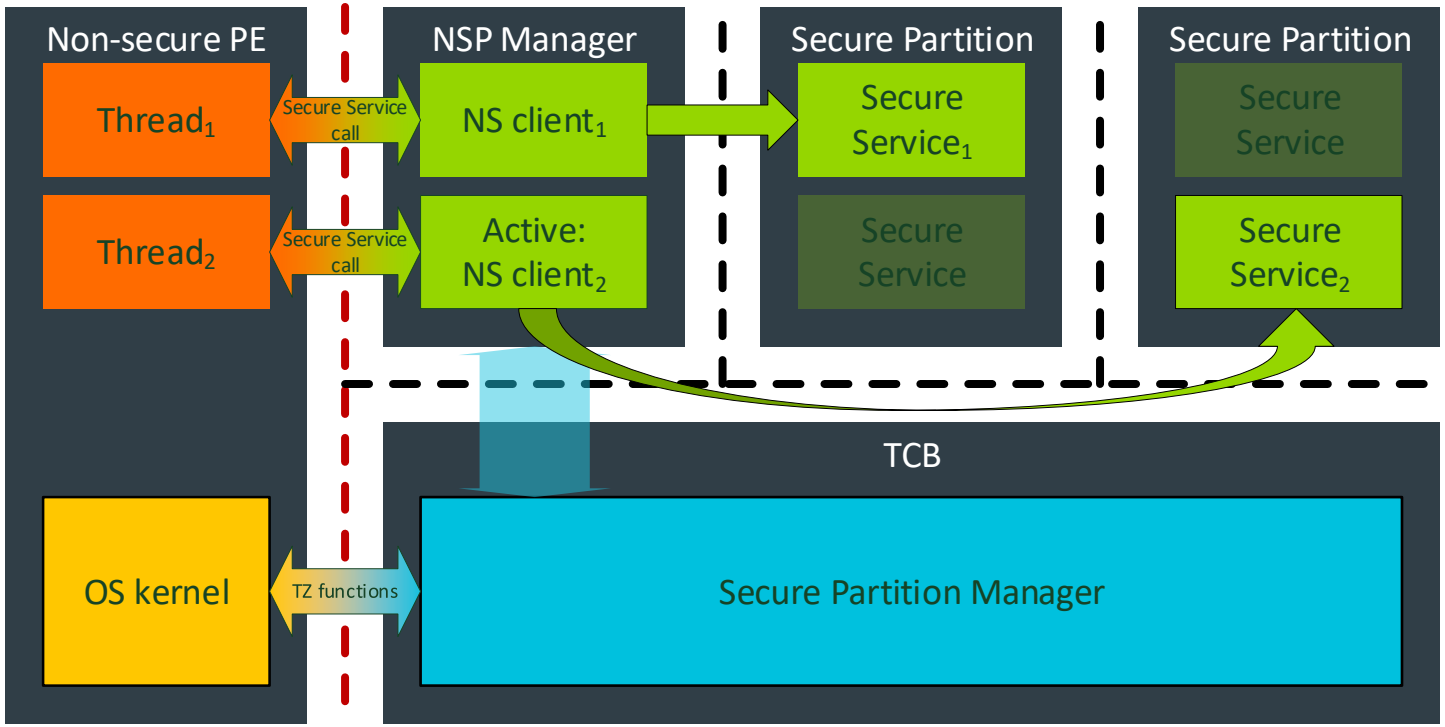
arm

Context Management Functions

Non-secure context awareness in Arm-v8M

1. **Non-secure threads created**
2. **Thread$_1$ calls Secure Service$_1$**
3. **Non-secure IRQ pre-empts operation -> context change**
4. **Thread$_2$ calls secure service$_2$**
5. **Secure service$_2$ returns**
6. **Thread$_2$ yields**
7. **Secure Service$_1$ returns**

NS RTOS provides notification to SPM about context creation, deletion, load or store operation, enabling NS context-dependent access to secure assets/services.



arm

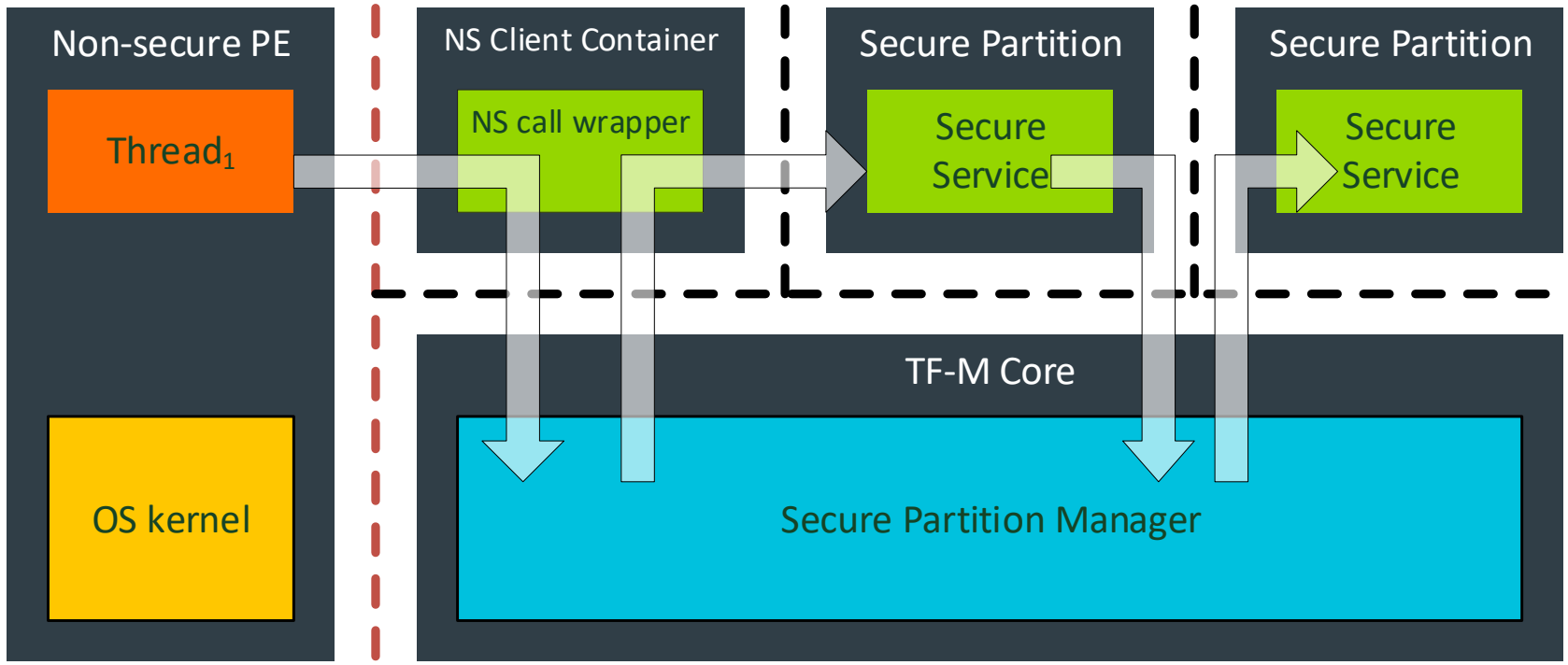
Implementations

arm

Trusted Firmware M library model

Secure Services implemented as functions

- Arm-v8M architecture support
- Secure Partition is a library of secure services
- Synchronous execution of secure services
- Programming model closely follows embedded/MCU concepts
- Low footprint – on demand allocation of resources

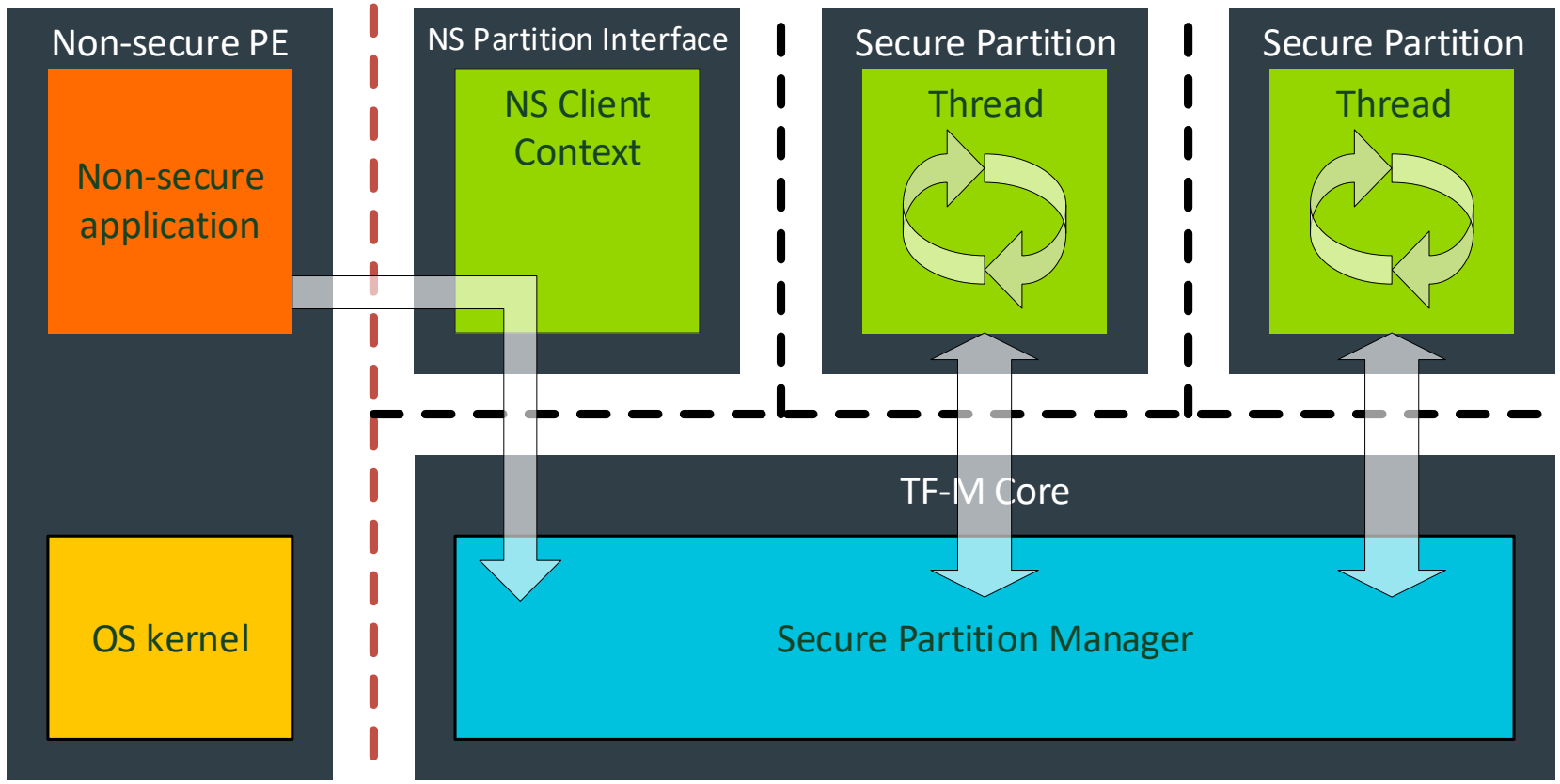


arm

Trusted Firmware M thread model

Secure Partitions implemented as threads

- Connection/message based interaction
- Robust, more prescriptive framework
- Static allocation of secure resources
- Asynchronous processing of service requests
- Less architecture dependent -> well suited for physical separation



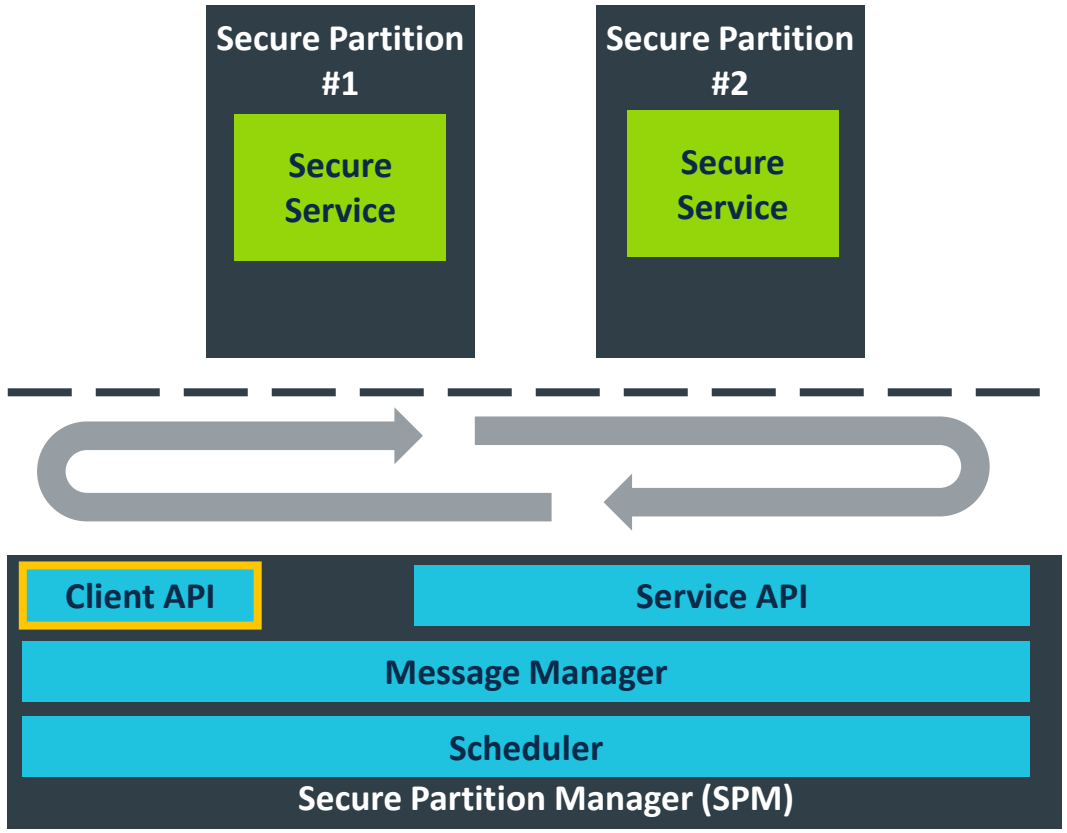
arm

Interaction with secure threads

arm

TF-M Inter-Process Communication (IPC)

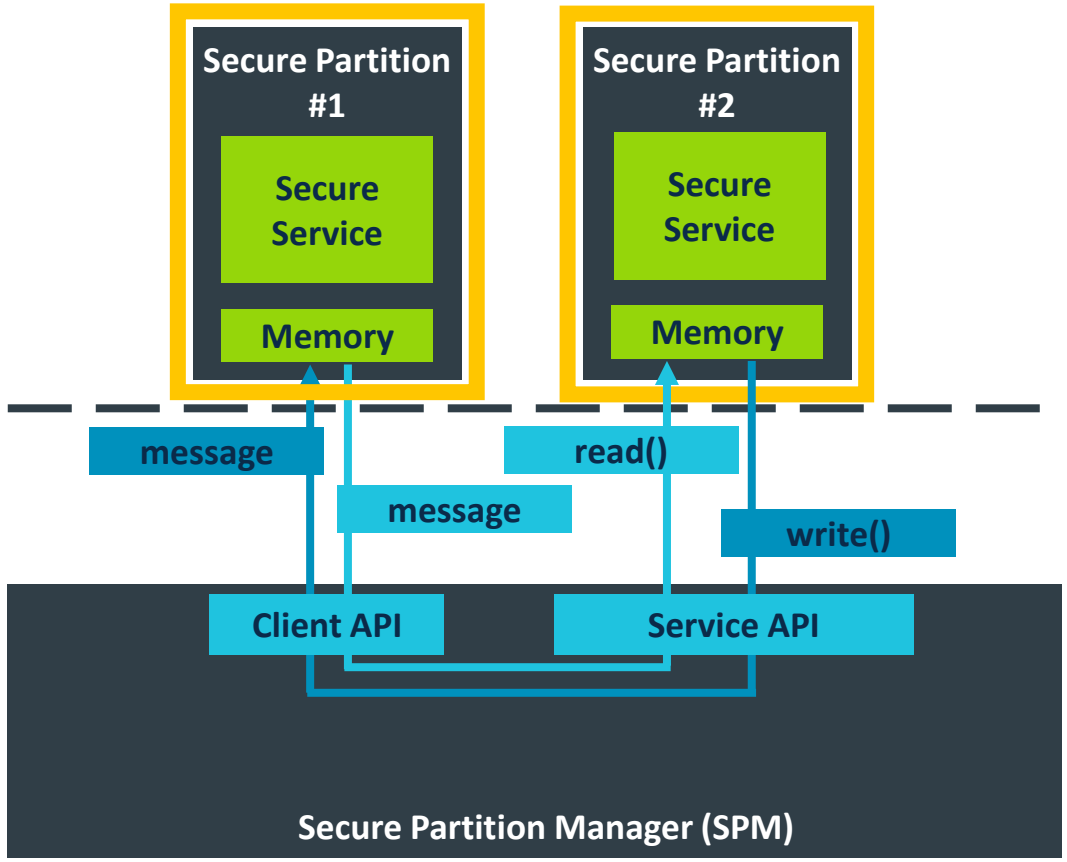
- **For TF-M Thread model**
- **Secure Partitions provide secure services**
  - NSPE is reflected as one Non-Secure Partition
- **One thread in one Secure Partition**
- **While loop in thread waiting for messages**
- **Client call sent as messages**
  - Non-Secure Partition is a client
  - Secure Partition could be a client
- **Service Interrupt is handled asynchronously**



arm

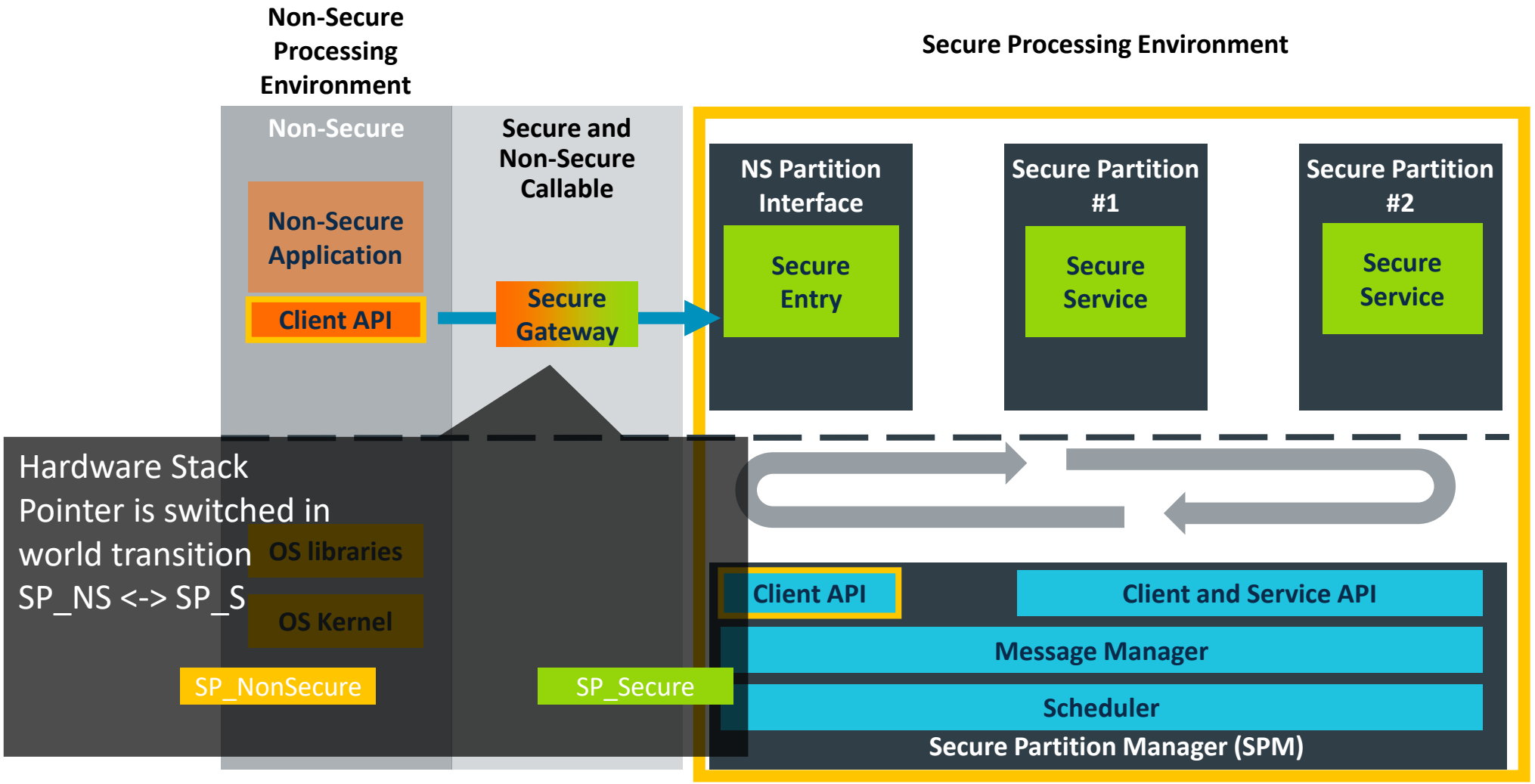
Security Consideration on Compartmentalization

- No shared memory
- Streamed read/write APIs for copying memory
- Memory integrity checking based on isolation level
- Peripheral usage is also isolated
- Runtime protection rule change to isolate



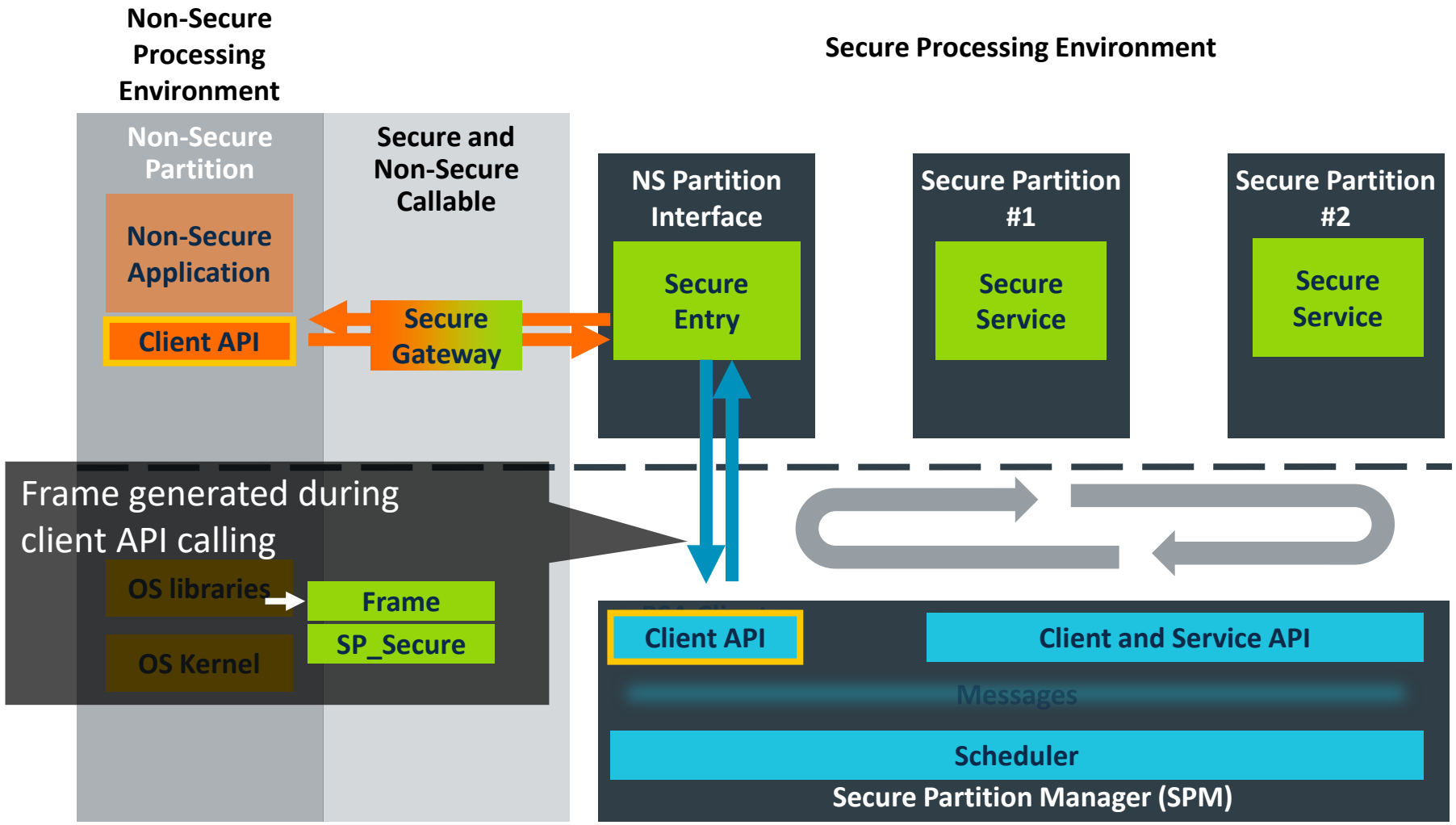
arm

Expand NSP with Arm-v8M TrustZone



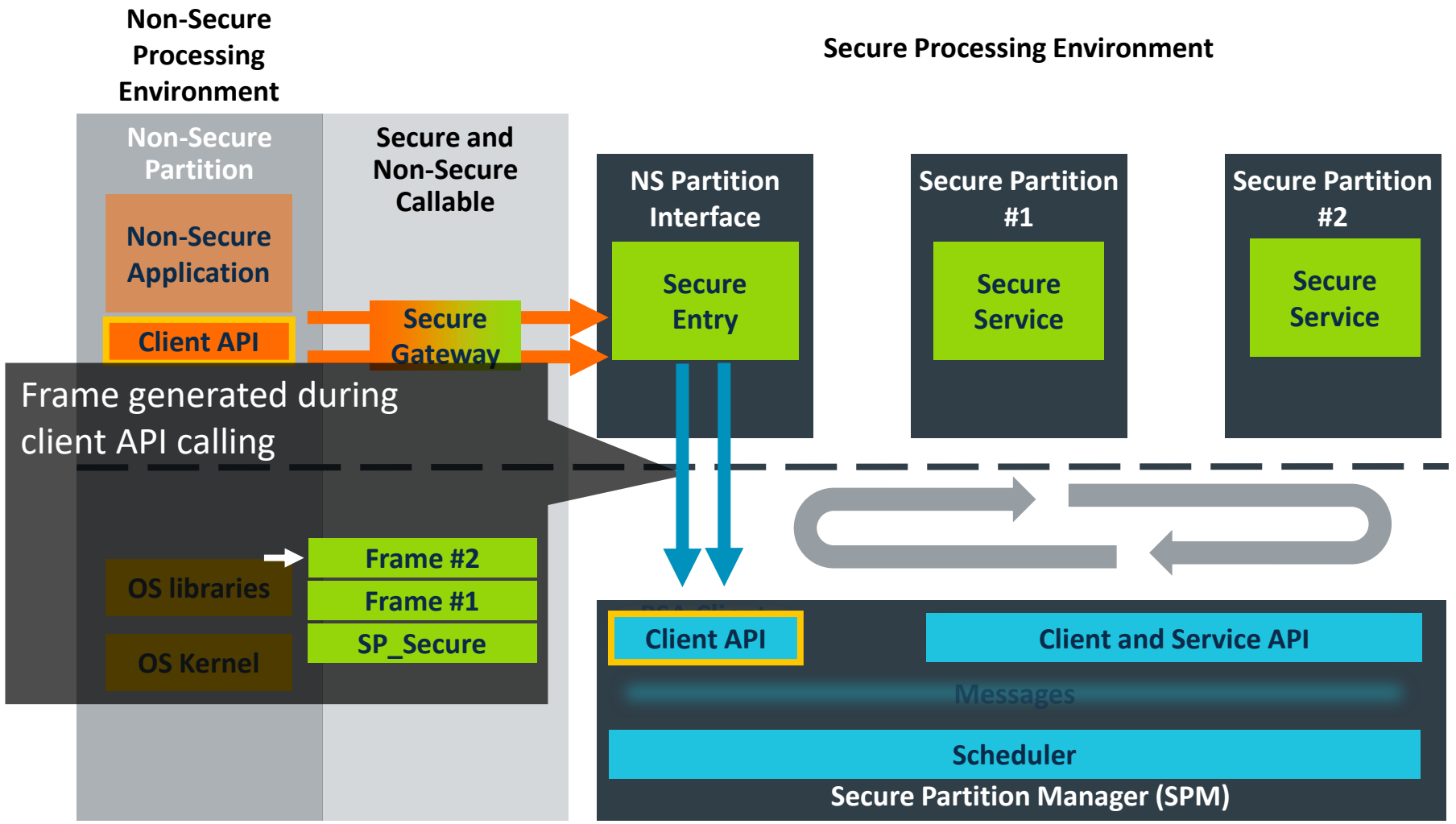
arm

Single NS Thread requests Secure Service



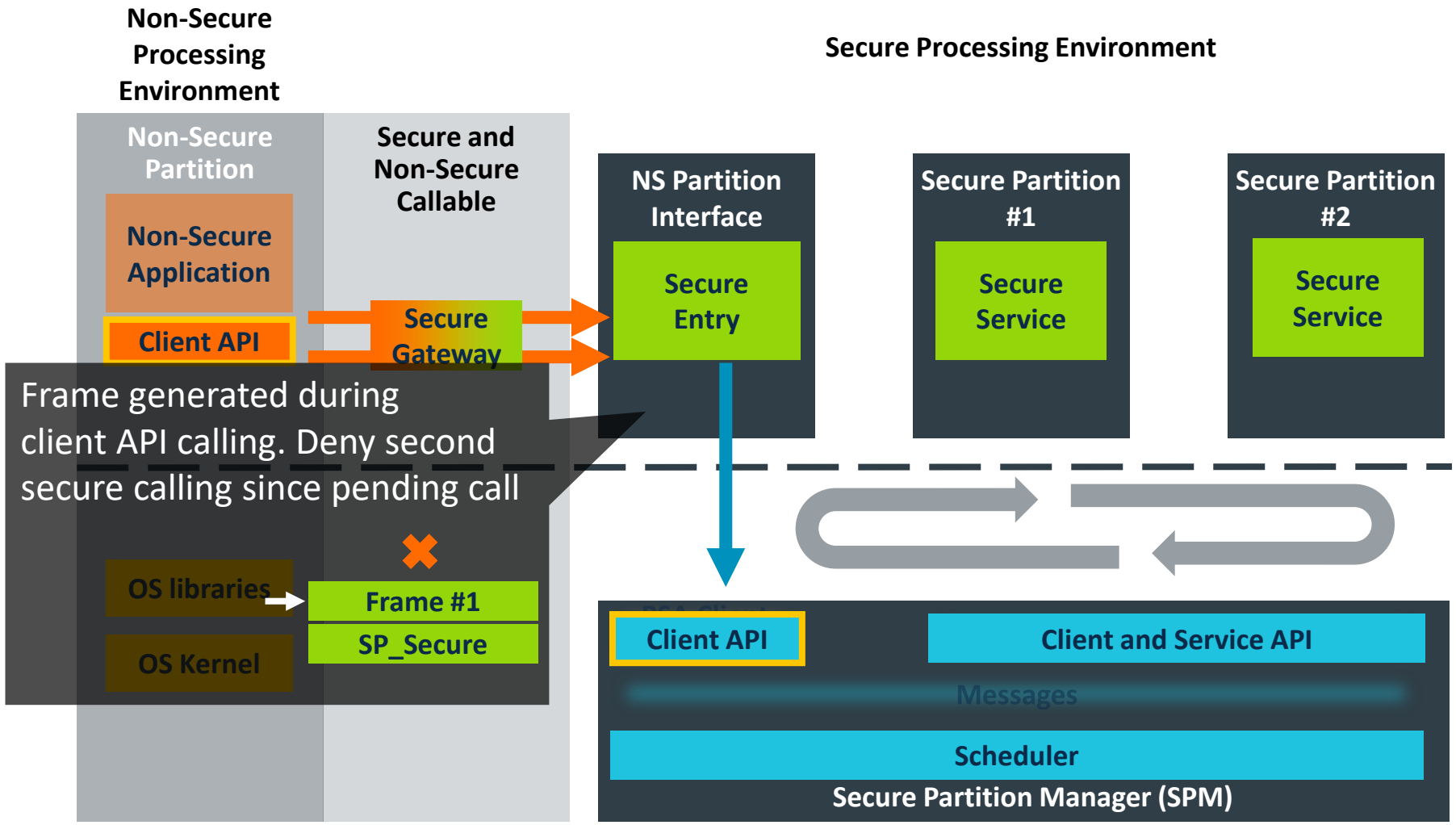
arm

Multiple NS Thread request Secure Service



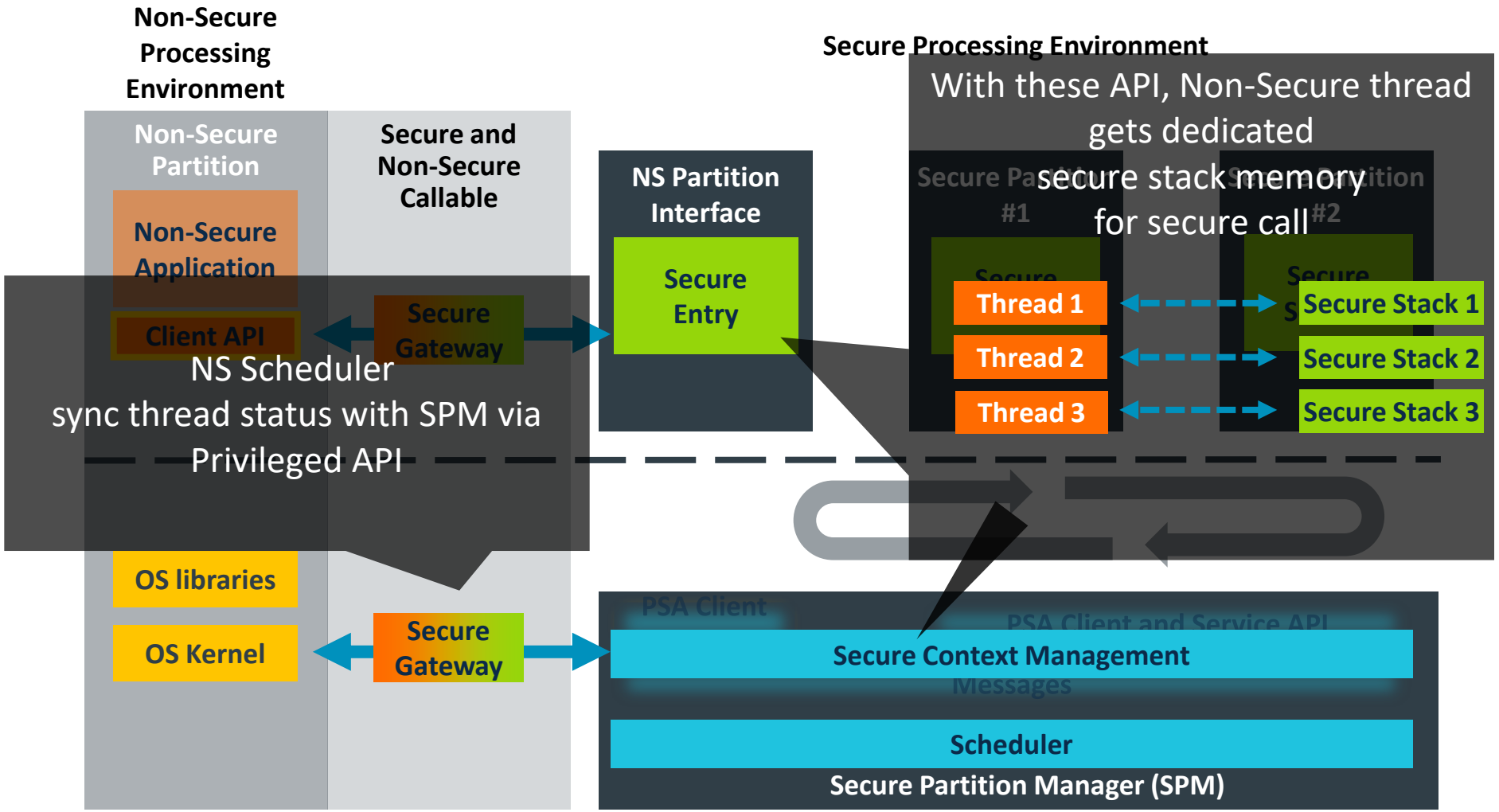
arm

Multi-Thread NSPE Secure Call Solution 1



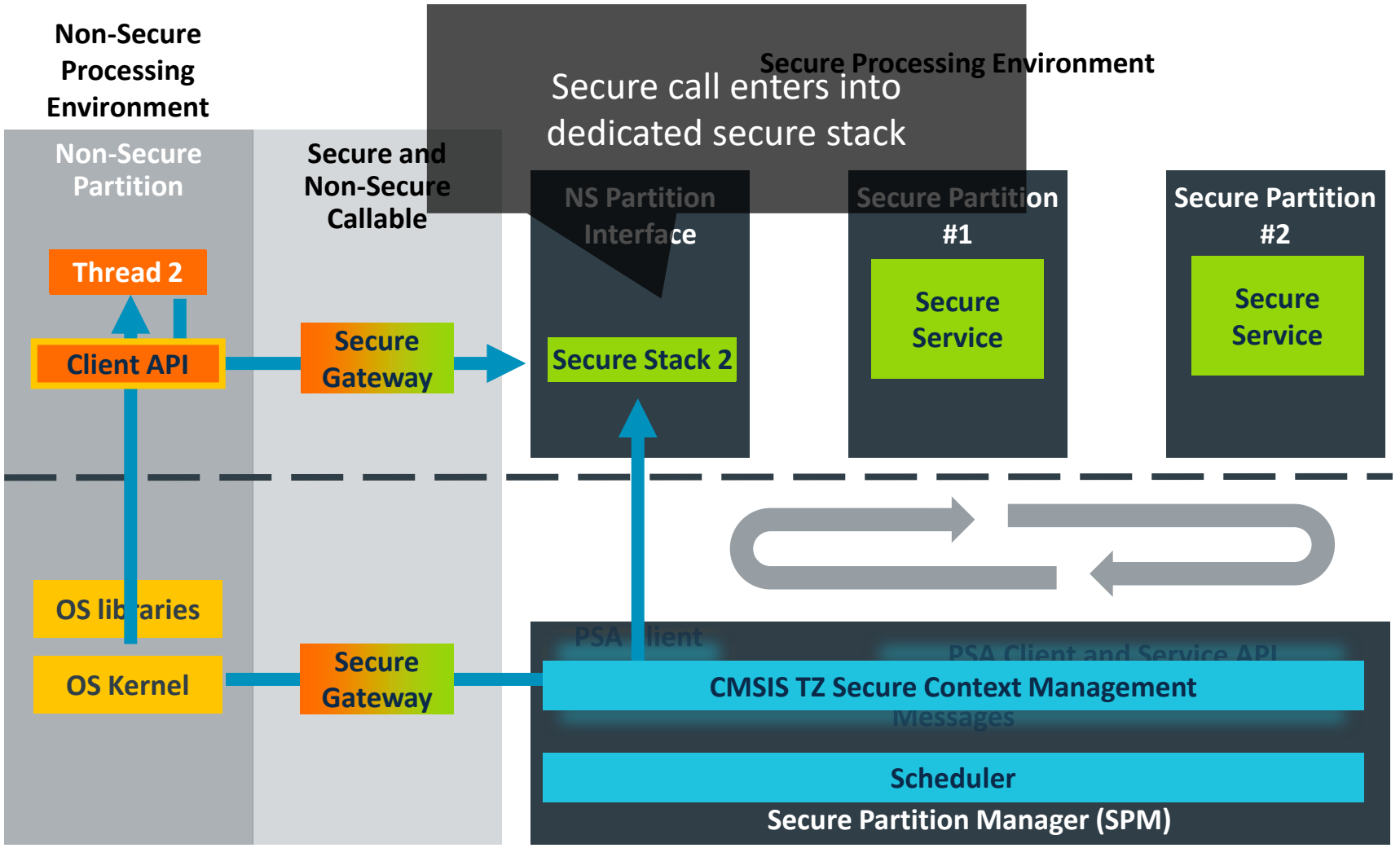
arm

Multi-Thread NSPE Secure Call Solution 2



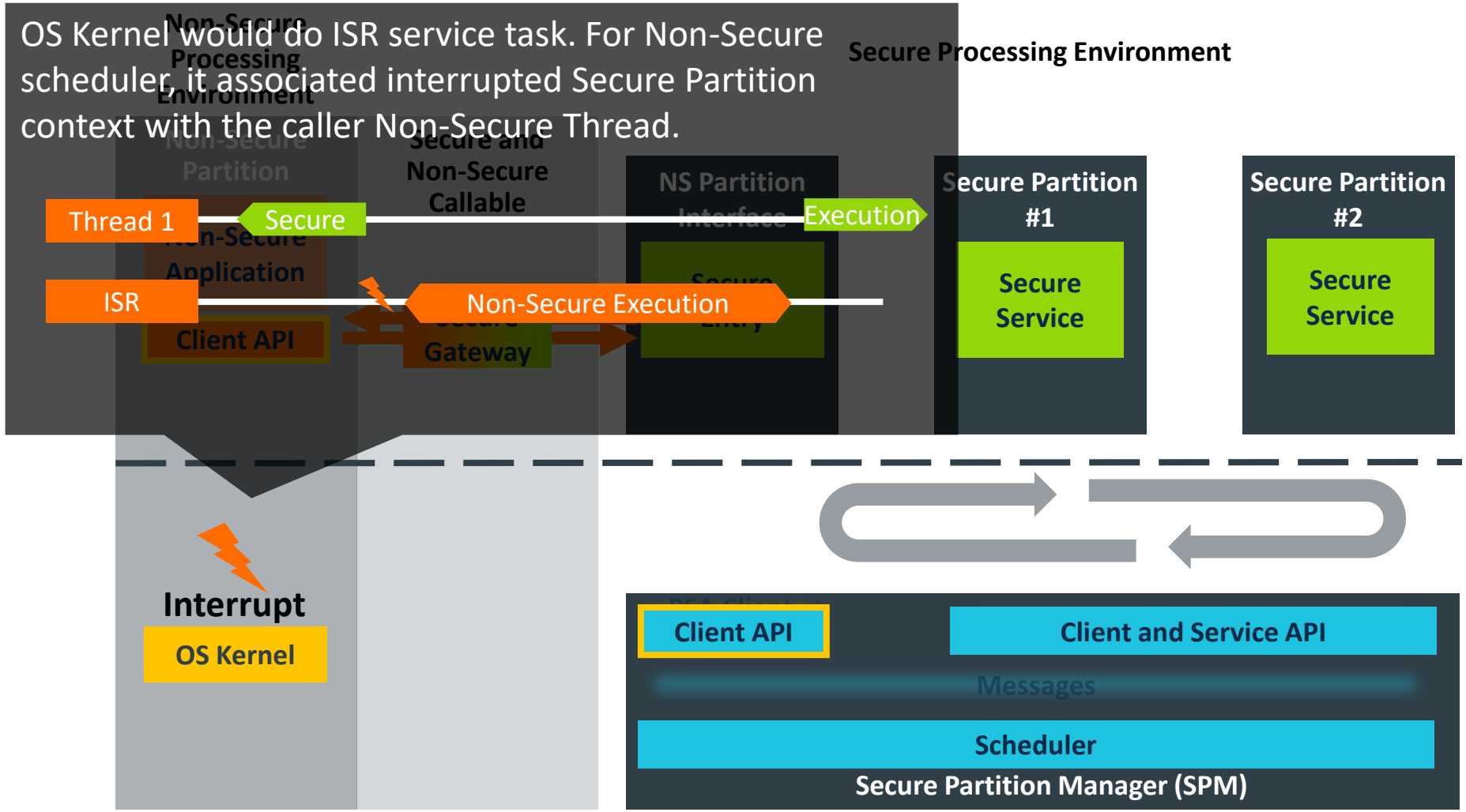
arm

Solution 2 Calling Process



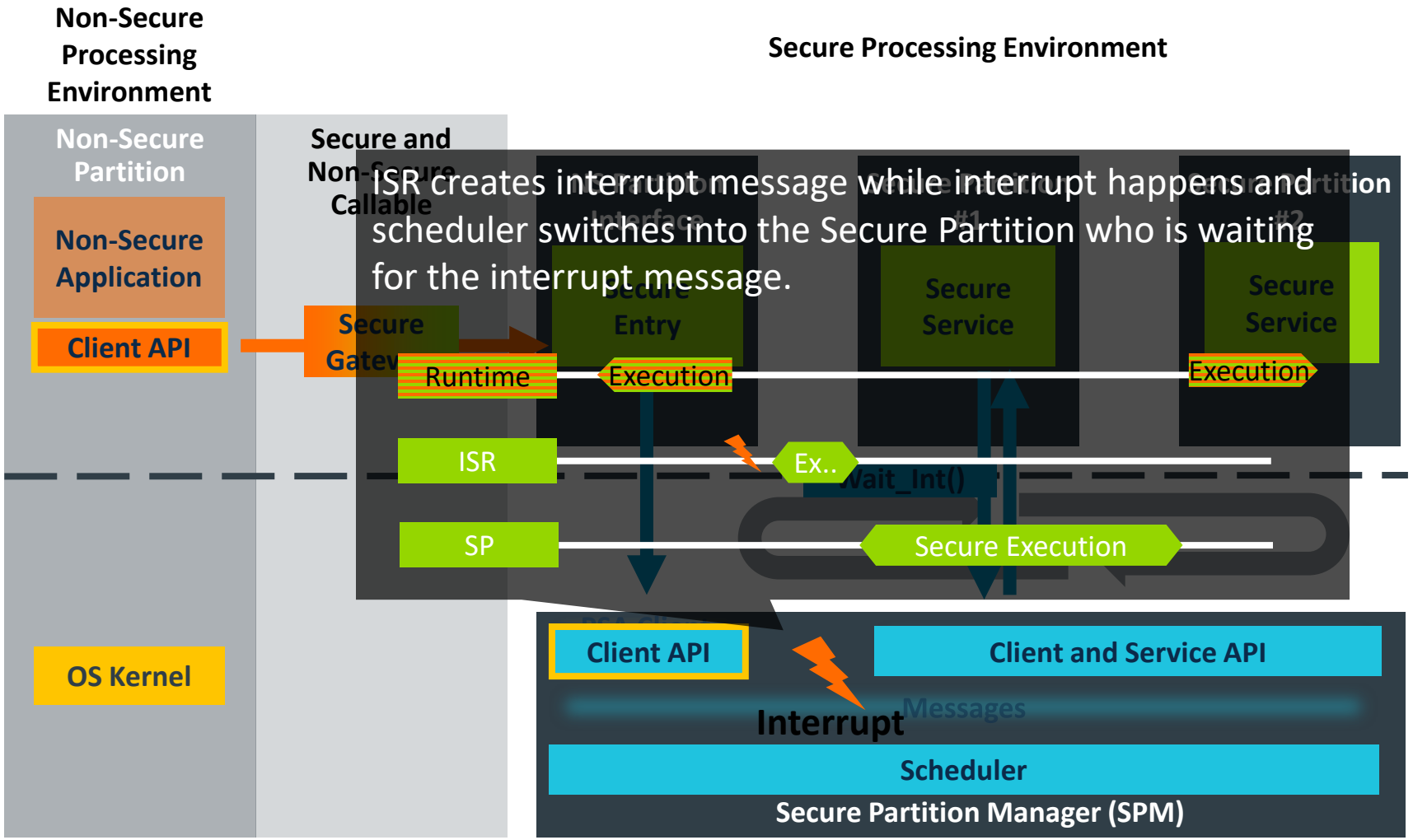
arm

Non-Secure Interrupt Preempts Secure Service



arm

Secure Interrupt Preempts Execution



ISR creates interrupt message while interrupt happens and scheduler switches into the Secure Partition who is waiting for the interrupt message.

arm

Summary

arm

Compartmentalization in IoT – No one-size-fits-all

Secure/non-secure isolation:
- physical
- temporal

Privilege control:
- none
- within secure domain
- within non-secure domain

Interaction:
- function calls
- IPC
- hardware mailbox

arm

How to get involved

TF-M is part of the Open Source/Open Governance trustedfirmware.org project

- Code base: https://git.trustedfirmware.org/

TF-M Team @ OpenIoT Summit Europe 2018

- Shebu Kuriakose
- Ashutosh Singh
- Ken Liu
- Miklos Balint

Get in touch

- Come round to the Arm booth during the summit
- Contact TF-M team at support-trustedfirmware@arm.com

More info on developer.arm.com and trustedfirmware.org

Thank You!
Danke!
Merci!
谢谢!
ありがとう!
Gracias!
Kiitos!
감사합니다
धन्यवाद