




# Confidential Compute for RISC-V Platforms

AP-TEE TG  
6/14/2022

*Assignee - Security HC*





# Disclosures



# 6/14 AP-TEE TG Agenda

- Disclosures
- Continue AP-TEE subtopics - Conf. Memory Tracking, Management
- Discuss spec development/topics (See next slide)

# AP-TEE Spec Review

Asciidoc Spec is at: <https://github.com/riscv-non-isa/riscv-ap-tee/tree/main/specification>

PDF version: <https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/riscv-aptee-spec.pdf>

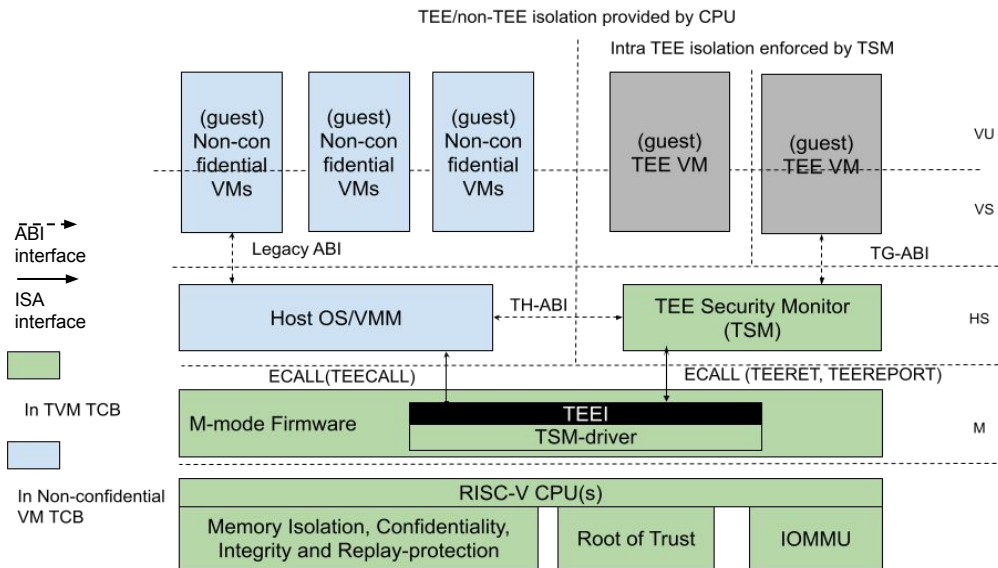
## Spec Sub-topics

- Arch Overview: <https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/overview.adoc>
- Threat Model -> <https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/threatmodel.adoc>
- Ref. Arch Details -> <https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/refarch.adoc>
- SW Lifecycle -> <https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/swlifecycle.adoc>
  - Conf. Memory Tracking -- see details in these slides 14 onwards.
- Attestation -> <https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/attestation.adoc>
- SBI extension -> [https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/sbi\\_atee.adoc](https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/sbi_atee.adoc)
- Interrupt Mgmt -> TBD
- Direct IO assignment to TVMs -> see details in these slides (text TBD)
- SW Model for Confidential process - > see ref. arch
- POC SW ratification process -> [Proposed ratification plan](#)

# AP-TEE TG Charter

## Define

- Reference architecture for confidential compute on RISC-V platforms
- AP-TEE TH/TG-ABIs -- normative non-ISA spec.
- AP-TEE Security Arch/Implementers Guide for RISC-V confidential compute - covers platform recommendations -- informative (living spec).
- AP-TEE ISA extension(s) -- *start with current ISA*; identify ISA gaps in TG -- request FT/TG as needed -- expected to be normative spec.



# (AP-TEE TG charter - 5/17 updates)

The RISC-V Application Platform - Trusted Execution Environment Task Group (AP-TEE TG) will collaborate to define the reference architecture for confidential computing on RISC-V **Application Processor-based** platforms **that support multi-tenant virtualized workloads**. The TG will define the ABI required to enable systems software to manage confidential workloads on a multi-tenant platform, while keeping the OS/hypervisor and entities that develop the OS/VMM and/or operate/manage the platform outside the TCB. The TG will design the interfaces to comprehend existing (ratified) privileged ISA and ensure extensibility of the interfaces to new Architectural ISA extensions as required for security or performance of confidential workloads. In addition to the normative specifications mentioned, the TG will produce an AP-TEE-specific security architecture analysis per the confidential computing threat model agreed upon as a living (non-normative) document supporting security recommendations, implementation-specific guidelines and relevant standard protocols for attestation for implementers of the AP-TEE capability on RISC-V platforms. The proposed RISC-V AP-TEE task group will collaborate to define these three specifications:

a. **AP-TEE reference architecture and SBI extension interface (non-ISA, normative)** which specifies the TH-ABI and TG-ABI interfaces to enable the OS/Hypervisor to manage confidential workloads on a multi-tenant platform, while keeping the OS/hypervisor and entities that develop the OS/VMM and/or operate/manage the platform outside the TCB.

The interfaces are defined between:

1. A new **RISC-V AP** platform-specific security service called the Trusted Security Manager (TSM) operating in RISC-V HS-mode and a general-purpose OS/Hypervisor executing in S/HS-mode - called the **TH-ABI**. The TH-ABI should cover aspects of: TVM creation and tear down, TVM measurement and attestation, TVM memory management and protection, TVM virtual-hart state management and protection, TVM execution and IO.

2. A Trusted Security Manager (TSM) running in HS-mode and a general-purpose OS executing in VS-mode - called the **TG-ABI**. The TG-ABI should cover aspects the TVM is involved in: TVM measurement extension and attestation, TVM memory conversion, TVM IO and other services used from host

b). AP-TEE architecture security analysis (**non-normative** living document) supporting recommendations and implementation guidelines for e.g. coverage of threat model, attestation protocols, crypto modes

c). AP-TEE **ISA extensions (normative)** to be proposed as needed for enforcing confidential workload security and performance requirements. **The baseline ISA expected for AP-TEE is the RISC-V privileged ISA (M, S/HS, U, VS modes), including the Hypervisor Extension**. The interfaces in item a. will be defined to be extensible to any such future ISA extensions. The TG will start with the definition of the programming interfaces and identify ISA gaps. ISA proposals made will be modeled via tools such as QEMU/Spike.

The goal of the AP-TEE interface specification is to **enable open-source reference implementations of the RISC-V AP-TEE interfaces** for platform-specific TSM implementations that enable confidential compute and trusted execution for different use case scenarios (Server, Automotive, Embedded etc.). To support this goal, a POC is defined that consists of: An SBI extension implementation for AP-TEE will be used as a reference implementation. A TSM implementation will be developed by the community as part of the ratification of the interfaces. The required changes will be made to the Linux/KVM host and guest software to validate the interface specifications.

# AP-TEE TG Charter: Interface specs

Specs



Area	Function	Resources
AP-TEE TH-ABI	SBI Extension Interface implemented by the TSM via ECALL for use by OS/VMM to manage TVMs	TG WG members
AP-TEE TG-ABI	SBI Extension Interface implemented by the TSM via ECALL for use by TVM guest workloads	
TEE Security Manager (TSM)	TSM is a RISC-V 64 bit SW module that uses RISC-V H-extension and implements TH and TG-ABI. It is in the TCB for all TVM workloads (Expected to be HW-vendor signed and may be HW-operator signed)	Rivos contributes to start collab.
M-mode FW	Minimal SBI extensions (TCB component) to support TSM initialization, TEECALL, TEERET implementation. It is in the TCB for all TVM workloads (Expected to be HW-vendor signed and may be HW-operator signed) - Collab with OpenSBI	Expecting collaborators on these existing projects from Software HC
Linux, KVM (Host OS/VMM)	<i>Untrusted</i> (enlightened) host OS/VMM that manage resources for TVM-based confidential workloads [TSM enforces security properties] - Collab with Hypervisor SIG	
Linux (TVM Guest OS), Guest Firmware	Enlightened guest OS/runtime (in TCB of TVM workload) - Collab with SW HC	

POCs



# AP-TEE TG Charter: Platform & ISA (Scope)

Area	Function	Resources
CPU	Evaluate AP-TEE mode qualifier, Sparse (page-based) confidential memory access-control	TG members
IOMMU	AP-TEE mode qualifier; Sparse (page-based) confidential memory access-control and fabric i/f	w/ IOMMU TG
TLB, Caches	AP-TEE mode qualifier and other micro-architectural structures	TG members
Interconnect, Fabric	Platform-specific cryptographic memory isolation and mode qualifier	TG members to document + Implementation feedback
Memory	Platform-specific cryptographic memory isolation and mode qualifier	
HW Root-of-trust	Platform-specific subsystem to support HW Attestation, Sealing interfaces	
Devices	Device-specific subsystem to support Device attestation, link security	
QoS, RAS, DC	Platform-specific, Domain-specific	w/ SOC Infra

**AP-TEE Security Arch for CC and Implementers Guide** covers recommendations on:

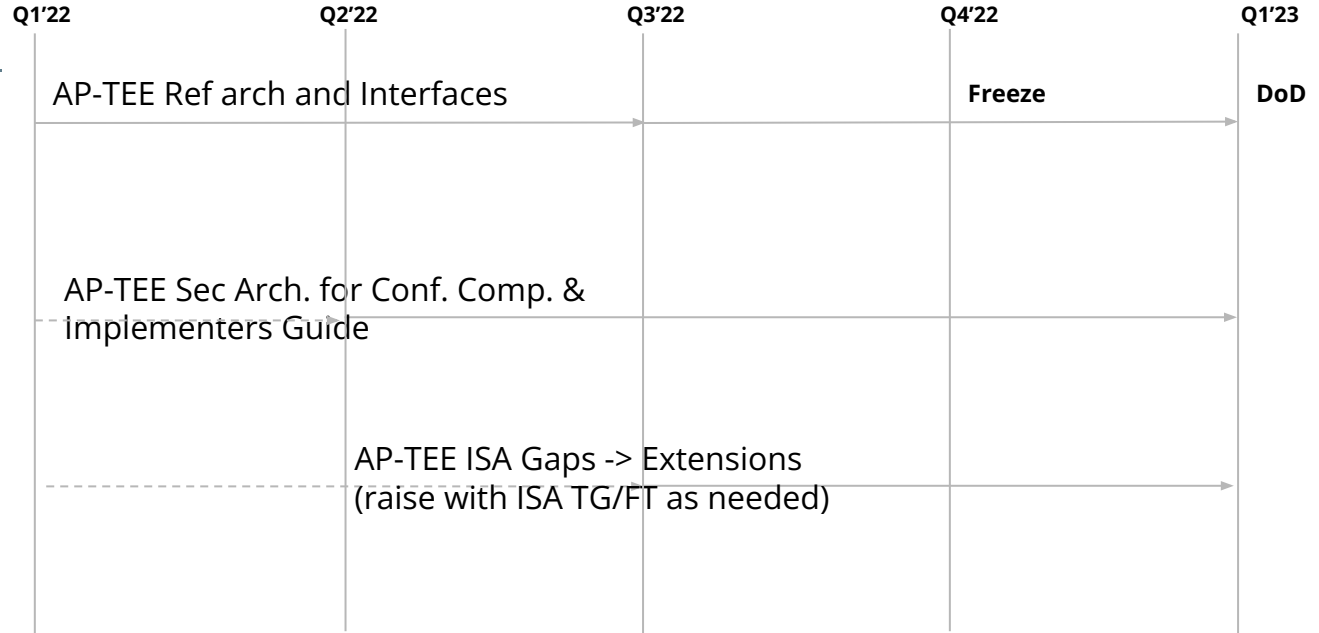
- Mapping of mitigations to threat model
- Recommendations for crypto modes
- Attestation protocols, formats



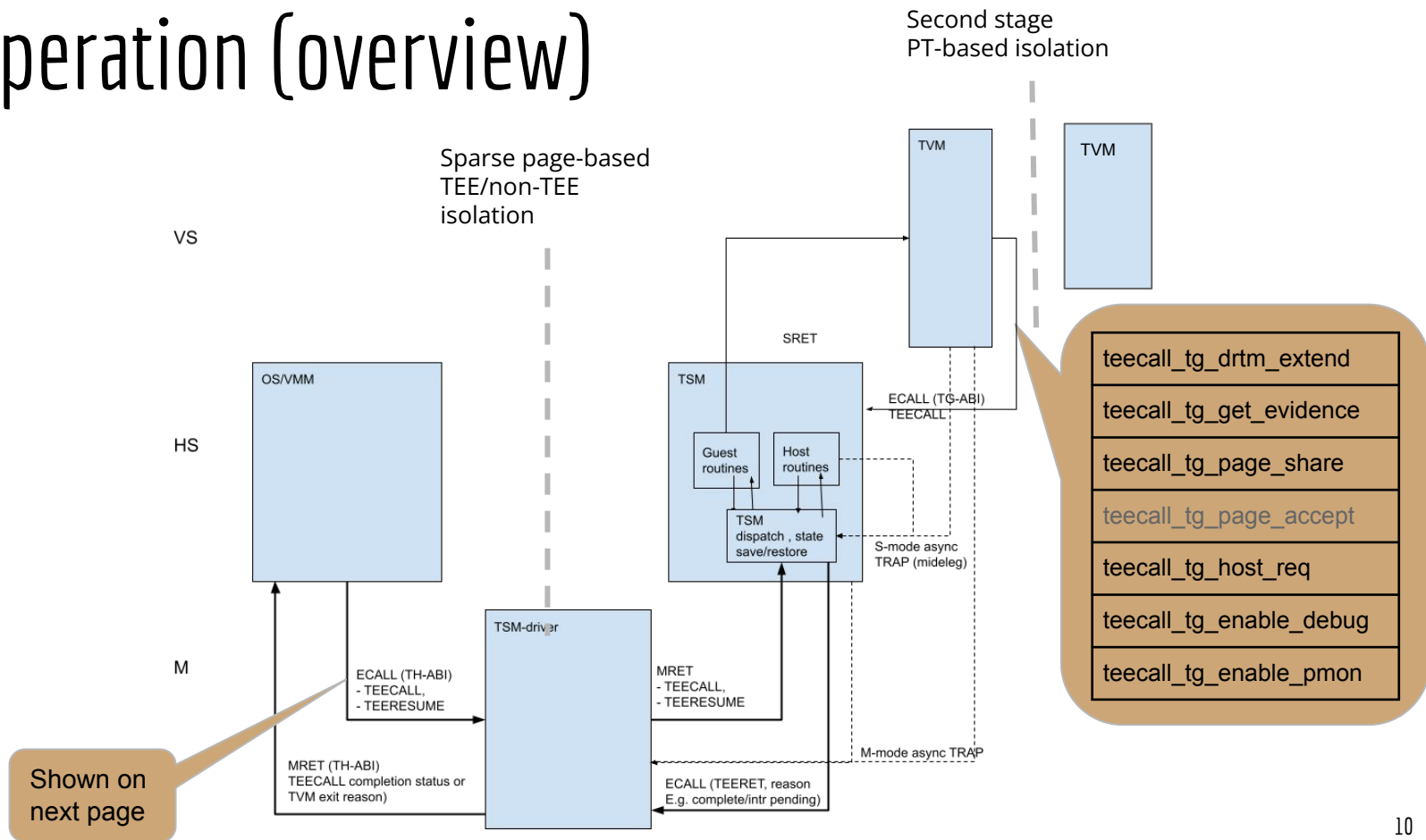
# AP-TEE TG workstreams

- [Proposed ratification plan](#)
- [DoD checklist](#)
- [Infra requirements](#)

- 
- Initial [AP-TEE spec](#) being discussed at the TC SIG.
  - [Confidential Computing Survey](#)



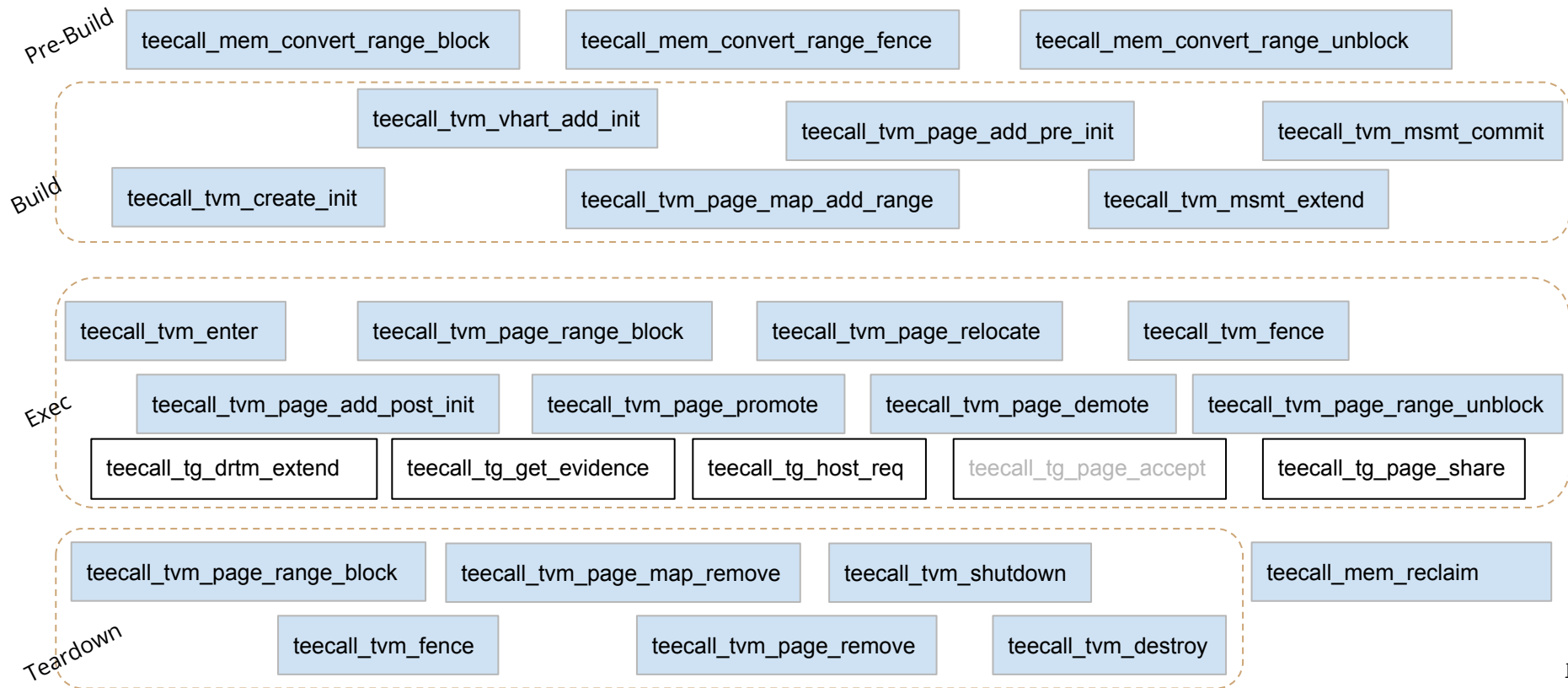
# TSM operation (overview)



# TVM Lifecycle

TBD:

- Add interrupt mgmt flows
- Add device assignment flows



# AP-TEE CPU Mode

- Drives access-control and isolation properties for TVM state stored in
  - Memory Isolation and Encryption
  - CPU caches
  - CPU registers
  - CPU micro-architectural state
- Enabled on a hart via a successful TEECALL <SBI call>
- Disabled on a hart via a successful TEERET <SBI call>
- Specify via new CSR - mapteectrl <priv ISA>

# AP-TEE Memory Isolation

A platform that supports AP-TEE requires the CPU to support new PMAs: **Confidential** and **Non-Confidential** [dynamic/programmable memory attributes]

A TVM needs to access two types of memory

- **Confidential memory** - has Confidential PMA - trusted used for TVM code, data
- **Non-Confidential memory** - has NC PMA - untrusted used for communication between TVM and untrusted entities

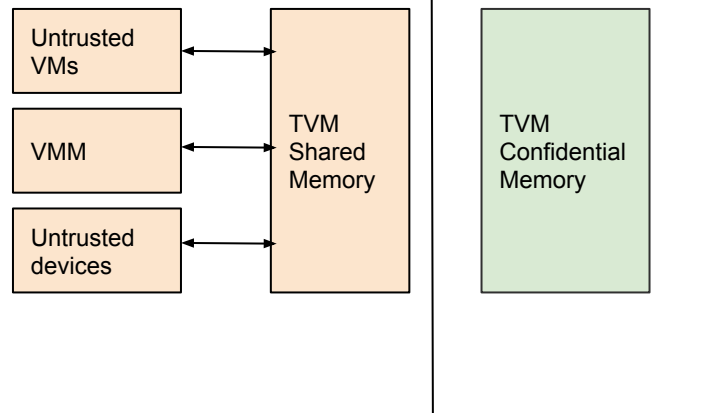
TSM provides TH-ABI primitives to the VMM to convert memory to Confidential [Assign] and vice-versa [Reclaim]

TVM memory is by default assigned from Confidential memory regions [enforced by the TSM]

TVM may indicate to convert memory assigned to Non-Confidential via TG-ABI (and back)

=> CPU AP-TEE mode =1 is allowed to access both Confidential and Non-Confidential memory

=> CPU AP-TEE mode =0 is allowed to access only Non-confidential Memory



# AP-TEE Confidential PMA Restrictions

Confidential PMA required for **implicit accesses** including:

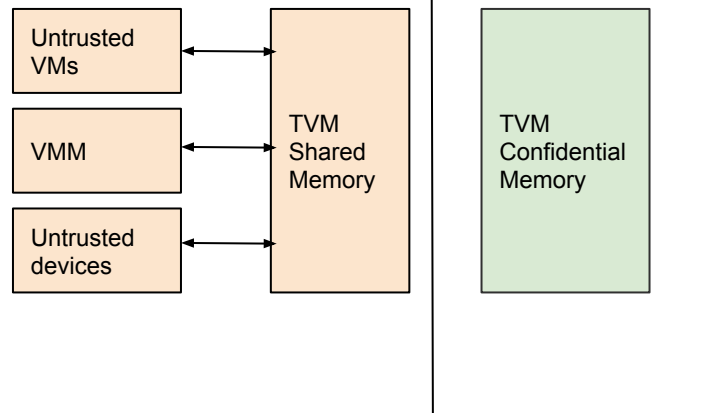
- Instruction fetch - security property: TVM cannot fetch code from untrusted shared memory
- First level paging structure walk - security property: TEE cannot locate page tables in untrusted shared memory

Memory with Confidential PMA may be associated with a unique memory encryption key (and similar IO/fabric protection policies)

Non-confidential memory is assigned by the VMM - the TSM and TSM-driver update the Confidentiality PMA by programming a

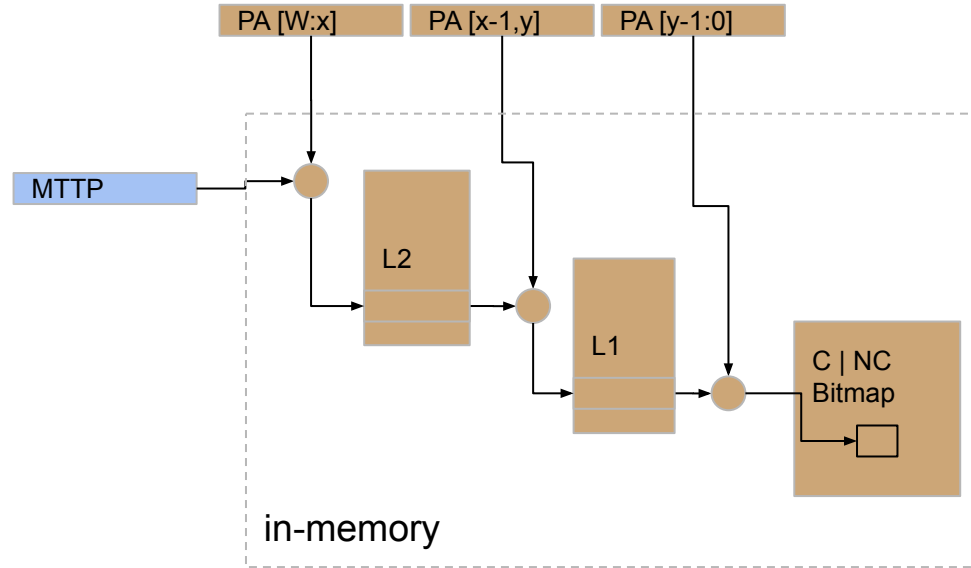
## **Memory Tracking Table**

TSM manages finer-granular (page-based) allocation from Confidential memory



# An Example Memory Tracking Structure

- Options: Flat or Hierarchical **in-memory** structure
  - Allows any physical addressable memory to be tracked
- Referenced by an MTTP CSR -- read/write accessible from TCB.
- When enabled, MTT looked up by memory physical address to determine confidentiality attribute of memory for implicit and explicit accesses
- Implementations may choose to cache MTT entries to optimize lookup



# Memory Tracking - Details

## Physical Memory Attributes

Hardware TCB enforced (MMU, IOMMU, *Memory Controller*)

SW TCB enforced (TSM-driver, TSM)

## Address Translation/Page Walk

Modes supported

## Handling Implicit, Explicit Accesses

- During TVM execution
- During TSM execution

## TLB management [covered in TH-ABI flows]

- Memory conversion (donation)
- Memory assignment (allocation)



# Physical Memory Attributes

Type: **Confidential** | **Non-Confidential** -- HW enforced

**Other attributes -- TSM enforced:**

Page-Type: Reserved | TSM | TVM

If Type is Confidential-TVM, additional attributes are to be tracked such as:

- TVM-owner
- Page-sub-type: HGAT | Data
- TLB version information
- Locking semaphore
- Additional meta-data
- etc.

**MTT** (walked by HW) dynamically maps PA in page sizes\* to Confidential | Non-Confidential memory attribute

**Extended MTT** - managed by the TSM (SW TCB) expected to maintain the remaining memory tracking information

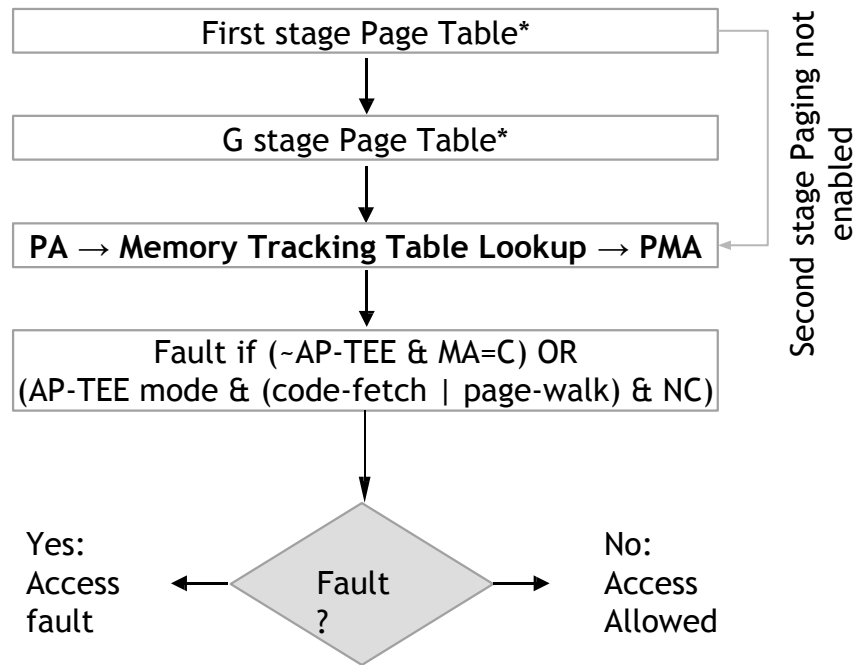
\*Since TVM and host/TSM assign memory in arch supp page sizes (512GB, 1GB, 2MB, 4KB)

# AP-TEE TVM Memory Access (Addr. Translation)

MTT provides PMA → Confidential or Not-Confidential (*dynamic*) attribute

TSM maps memory pages to specific TVMs (via G-stage translation) - pages may belong to C|NC PMA

\*See RISC-V [priv-isa spec](#) for paging modes



# Implicit, Explicit Accesses

TVM/TSM implicit accesses:

- First level paging structure walk - security property: TVM/TSM cannot locate page tables in non-confidential memory

TSM implicit accesses:

- G-stage paging structure walk - security property: TSM cannot locate G-stage structure in non-confidential memory

TSM/TVM code fetch:

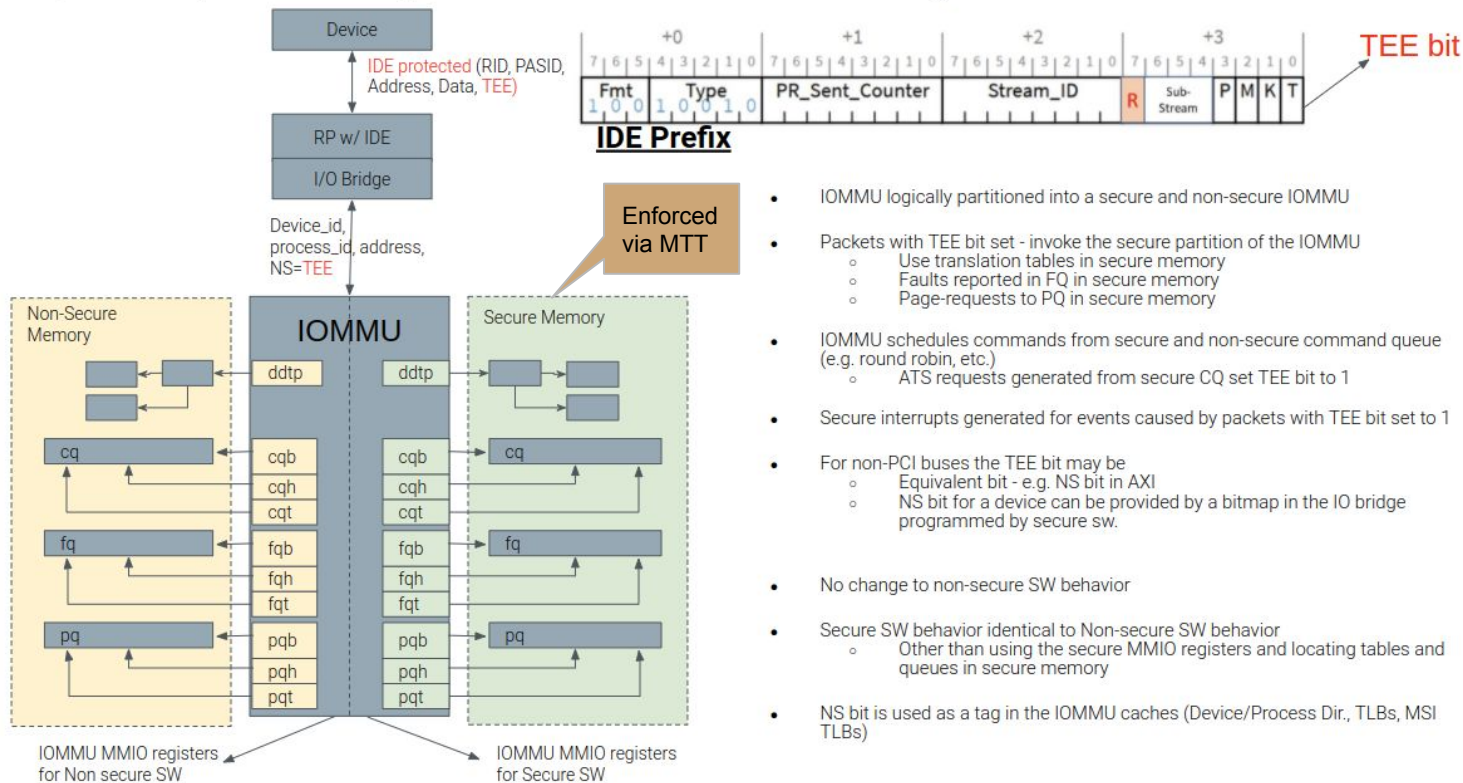
- Instruction fetch - security property: TVM/TSM cannot fetch code from untrusted shared memory

TSM/TVM data fetch:

- Data fetch - security property: TVM/TSM allowed to relax data accesses to non-confidential memory (via MTT)

All confidential memory (Memory with Confidential PMA) may be associated with a memory encryption key (and similar IO/fabric attributes to drive access-control, encryption policies)

# IOMMU Implications

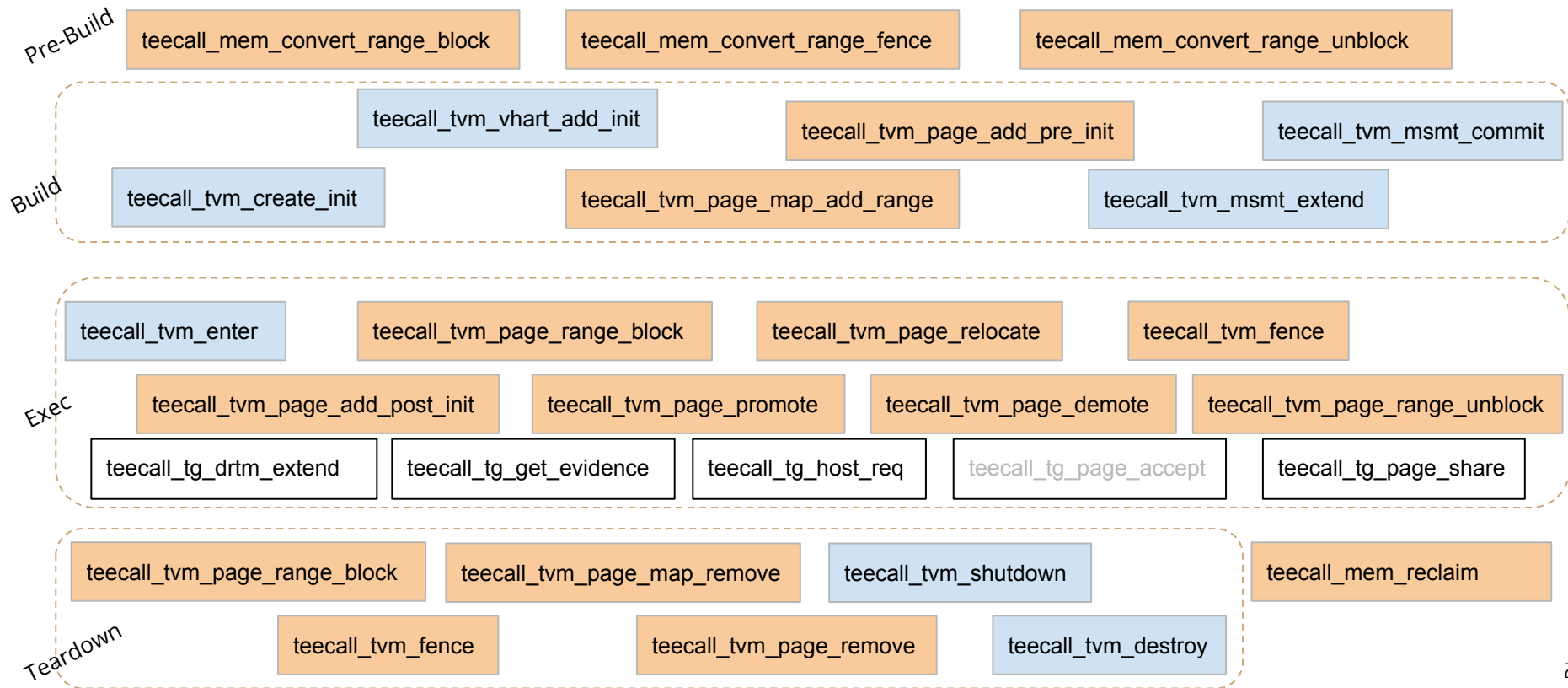


- IOMMU logically partitioned into a secure and non-secure IOMMU
- Packets with TEE bit set - invoke the secure partition of the IOMMU
  - Use translation tables in secure memory
  - Faults reported in FQ in secure memory
  - Page-requests to PQ in secure memory
- IOMMU schedules commands from secure and non-secure command queue (e.g. round robin, etc.)
  - ATS requests generated from secure CQ set TEE bit to 1
- Secure interrupts generated for events caused by packets with TEE bit set to 1
- For non-PCI buses the TEE bit may be
  - Equivalent bit - e.g. NS bit in AXI
  - NS bit for a device can be provided by a bitmap in the IO bridge programmed by secure sw.
- No change to non-secure SW behavior
- Secure SW behavior identical to Non-secure SW behavior
  - Other than using the secure MMIO registers and locating tables and queues in secure memory
- NS bit is used as a tag in the IOMMU caches (Device/Process Dir., TLBs, MSI TLBs)

# TVM Lifecycle

TBD:

- Add interrupt mgmt flows
- Add device assignment flows



# Page Conversion (TH-ABI)

teecall\_mem\_convert\_range\_block

teecall\_mem\_convert\_range\_fence

teecall\_mem\_convert\_range\_unblock

Post measured-boot, TSM must know the addressable Confidential memory on the platform

- Memory region used for the TSM are setup in the MTT by the TSM-driver
- the TSM manages the MTT from init onwards.

Page conversion involves the following steps by the TSM:

- Verify page(s) donated by the VMM is/are Non-Confidential page(s)
- Create **blocked MTT entries** (synchronized) for the requested page(s) and size as Confidential pages
- TSM enforces a **TLB version scheme** and enforces that the VMM perform the **invalidation** of the hart TLBs (via IPIs) to remove any cached mappings
  - Pages may be scrubbed later since page contents may be initialized before assignment to a TVM
- VMM completes the donation of selected pages by **unblocking MTT entries** - TSM verifies that TLB fence (c) was completed for the pages selected for conversion
- At this point non-TEE mode software cannot create new TLB entries to donated pages - since non-TEE mode accesses to MTT-tracked Confidential pages will fault (including implicit accesses)

# Page Assignment - Security invariants

teecall\_tvm\_page\_map\_add\_range

The following are the security requirements/invariants for enforcement of the memory access-control for memory assigned to the TVMs. These rules are enforced by the TSM and the HW:

1. Contents of a TVM page assigned (statically measured or lazy-initialized) to the TVM is bound to the Guest PA assigned to the TVM during TVM operation. [TSM-enforced]
2. Single owner - a TVM page can only be assigned to a single TVM, and mapped via a single GPA unless aliases are allowed in which case, such aliases must be tracked by the TSM). Aliases in the virtual address space are under the purview of the TVM OS. [TSM-enforced]
3. 1st stage address translation - A TVM page mapping must be translated only via first stage translation structures which are contained in confidential type pages, and assigned to the same TVM [HW, TSM-enforced].
4. 2nd stage address translation [TSM-enforced]:
  - a. A TVM page guest physical address mapping must be translated only via the TSM-managed second stage translation structures for that TVM.
  - b. 2nd stage structures may not be shared between TVMs, and must not refer to any other TVMs pages.
  - c. The OS/VMM has no access to TVM second stage paging structures (or ciphertext) [HW-enforced]
  - d. Circular mappings in the second stage paging structures are disallowed.
5. Access to shared memory pages must be explicitly signaled by the TVM and enforced for memory tracking (via TSM, CPU).

# Page Assignment (TH-ABI)

teecall\_tvm\_page\_map\_add\_range

VMM uses this operation to add a hgat structure page to be used for mapping a guest physical address (GPA) to a supervisor physical address (SPA). The inputs to this operation are:

- TVM identifier
- Supervisor physical address(es) for the new page(s) to be used for the hgat structure entries
- GPA and hgat level to be used for the mapping to be added

TSM Operation:

1. Verify via the TVM has been created
2. Verify that the PPN(s) for the new page(s) to be used for TVM hgat is/are Unassigned-Confidential per the MTT
3. For the GPA to be mapped, perform a TVM-hgat walk to locate the non-leaf entry that should refer to the new page being added (to hold the next level of the mapping for the GPA). If the mapping already exists, the operation is aborted.
4. Initialize the new hgat page to zero (no hgat page table entries are valid)
5. Update the parent hgat entry to refer to the new hgat page (mark non-lead as valid)
6. Update the hgat page EMTT entry with the TVM owner-id and page-type



# Page Add - Measured (TH-ABI)

teecall\_tvm\_page\_add\_pre\_init

The `page_map_add` creates a non-present GPA mapping for the TVM

This intrinsic (`page_add_pre_init`) makes the page mapping valid for the TVM with a reference to a page w/ measured contents

## TSM Operation:

1. Verify that the TVM is valid
2. A source page must be provided, this operation can only be performed if the TVM measurement has not been finalized. (pre-init stage).
3. Verify that the PPN for the new (dest) page to be used for TVM is Confidential-unassigned in the Extended MTT
4. For the GPA to be mapped, perform a TVM-hgat walk to locate the leaf entry that should refer to the new page being added . If the mapping does not exist OR exists but is not in the valid expected state, the operation is aborted.
5. Initialize the new TVM page with contents from source page. Note that the initialization of memory will be in AP-TEE-mode and via the TSM-owned virtual mapping of the page assigned to the TVM.
6. The measurement of the TVM is extended with the GPA used to map to the dest page.
7. Update the TVM page EMTT entry to be Confidential-Assigned with the TVM owner and page type as TVM-Data
8. Update the leaf hgat page table entry to refer to the new page (mark leaf as valid) to allow future TLB mappings to be created when the TVM is executing (subsequently).

# Page Add - Lazy/Zero (TH-ABI)

teecall\_tvm\_page\_add\_post\_init

The page\_map\_add creates a non-present GPA mapping for the TVM

This intrinsic (page\_add\_post\_init) makes the page mapping valid for the TVM with a reference to a zero (confidential) page

## TSM Operation:

1. Verify that the TVM is valid
2. No source page is provided, this operation can only be performed if the TVM measurement has been finalized. (TVM must be in the post-init stage).
3. Verify that the PPN for the new page to be used for TVM is Confidential-unassigned in the Extended MTT
4. For the GPA to be mapped, perform a TVM-hgat walk to locate the leaf entry that should refer to the new page being added . If the mapping does not exist OR exists but is not in the valid expected state, the operation is aborted.
5. Initialize the new TVM page with contents from source page OR zero if no source page is provided (for lazy addition of memory to TVM). Note that the initialization of memory will be in AP-TEE-mode and via the TSM-owned VA mapping of the page assigned to the TVM.
6. If the source page contents are provided, the measurement of the TVM is extended with the GPA used to map to the page.
7. Update the TVM page EMTT entry to be Confidential-Assigned with the TVM owner and page type as TVM-Data
8. Update the leaf hgat page table entry to refer to the new page (mark leaf as valid) to allow future TLB mappings to be created when the TVM is executing (subsequently).

# TVM TLB mgmt (TH-ABI)

If the VMM requires to make any change to an assigned and present GPA mapping (e.g. remove, relocate, promote etc. ) - then it must execute the following sequence (enforced by TSM) to affect that change:

1. **Block the mapping it wants to modify (page or range of pages)**
  - a. Prevents new mappings from being populated in the TLB (and requires unblock)
  - b. in the PA metadata maintained by the TSM, captures the `tvm_tlb_tracker` version into per page `pg_tvm_tlb_tracker`
2. **Initiate Fence/increment the `tvm_tlb_tracker` version for the TVM (on any one hart)**
  - a. Checks that the previous TLB synchronization was completed [`refcount` corresponding to `tvm_tlb_tracker` is 0], and then increments the `tvm_tlb_tracker` counter (Only previous and current `refcount` has to be maintained)
3. **VMM must issue an IPI to each TVM virtual-hart executing to trap to the TSM -- enables the TSM to perform a Hfence.GVMA**
  - a. TVM exit/trap decrements the `refcount` for active TVM virtual-harts [reported to VMM as normal]
  - b. This step prevents pre-existing (stale) mappings from being utilized
  - c. *A (subsequent) virtual-hart being entered compares the `tvm_tlb_tracker` with the `vhart_tlb_tracker` version and on a mismatch is subject to a flush during TVM Enter (and brings `vhart_tvm_tlb_tracker` up to date with the `tvm_tlb_tracker`)*
4. **-----No active/usable translations for affected TVM 2nd stage mappings after this point-----**

`teecall_tvm_page_range_block`

`teecall_tvm_fence`

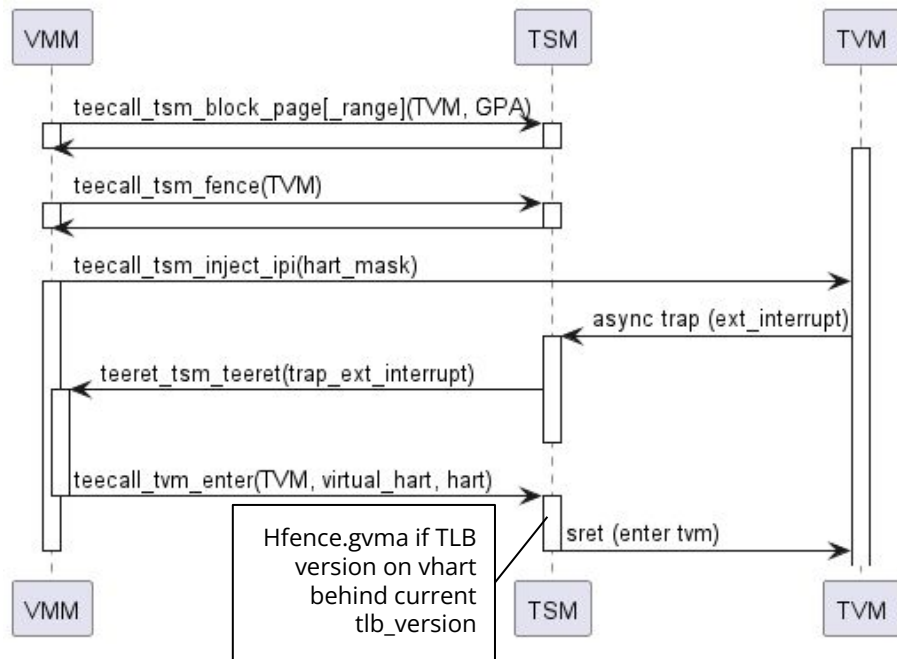
`teecall_tvm_page_<op>`

...

`teecall_tvm_page_range_unblock`

# TVM TLB mgmt/2 (TH-ABI)

5. Invoke the specific **mapping change** operation (remove, relocate, promote, migrate etc.)
  - a. This step checks that the affected mapping(s) are blocked in 2nd stage mapping
  - b. and, uses the recorded PA metadata `pg_tvm_tlb_tracker` to check that the TLB synch in step 3 was completed (`refcount=0`)
6. **Unblock** the mapping(s)
  - a. Checks that TLB flushing step 3 has completed for TVM virtual-harts, then unblocks the 2nd stage mapping (& clears PA blocking meta-data)



# Page Removal (TH-ABI)

teecall\_tvm\_page\_range\_block

teecall\_tvm\_fence

teecall\_tvm\_page\_map\_remove

1. **VMM must ensure no TVM virtual-harts are executing**
  - a. **this property is enforced by the TSM before the page remove operation is allowed to succeed**
2. **VMM participates in the page\_range\_block for the selected GPA range it wants to remove**
3. **TSM verifies the page range under action is blocked and TLB versioning is performed as described in the slides 22-23**
4. **-----No active/usable translations for affected TVM 2nd stage mappings after this point-----**
5. **VMM may remove the mappings successfully via the page\_map\_remove**
  - a. This operation puts the removed page and the hgat mapping structure pages in a Confidential-Unassigned state.
  - b. VMM may use those pages for other TVMs (or may reclaim those pages via teecall\_mem\_reclaim)

# Page Promote (TH-ABI)

teecall\_tvm\_page\_promote

1. **VMM selects a GPA page range to be promoted to a large page mapping (4KB->2MB or 2MB->1GB etc.)**
2. **VMM participates in the page\_range\_block for the selected GPA range**
3. **TSM verifies the page range under action is blocked and TLB versioning performed as described in the slides 22-23**
4. **-----No active/usable translations for affected TVM 2nd stage mappings exist after this point-----**
5. **VMM invokes the page promote intrinsic for the gpa\_range**
  - a. **TSM verifies the TVM is valid**
  - b. **TSM verifies that all PA fragments under the specified GPA range are consistently mapped (present) and with same permissions before the page promote operation is allowed to proceed**
  - c. **TSM updates large page PTE to reference first large\_page aligned PA in range**
  - d. **TSM returns the freed hcat PTE 4KB PFN as an Confidential-Unassigned page to the VMM**
  - e. **VMM may use that page for other TVMs (or may reclaim those pages via teecall\_mem\_reclaim)**
6. **Subsequent TLB mappings for the gpa range will create large page mappings**

# Page Demote (TH-ABI)

teecall\_tvm\_page\_demote

1. **VMM selects a GPA page to be demoted to a set of small page mappings (1GB->2MB or 2MB->4KB etc.)**
2. **VMM participates in the page[\_range]\_block for the selected GPA (range)**
3. **TSM verifies the page range under action is blocked and TLB versioning is performed as described in the slides 22-23**
4. **-----No active/usable translations for affected TVM 2nd stage mappings exist after this point-----**
5. **VMM invokes the page demote intrinsic for the gpa\_range, specifying an additional PA to use for the additional HGAT level PTE page**
  - a. **TSM verifies the TVM is valid**
  - b. **TSM verifies that PA for the specified large GPA page is consistently mapped (present)**
  - c. **TSM updates PTEs for new GHAT page with small GPA page references and with same permissions before the page demote operation is allowed to proceed**
  - d. **TSM updates the previous PTE to reference new HGAT PTE page**
6. **Subsequent TLB mappings for the gpa range will create (fragmented) small page mappings**

# Page Reclaim (TH-ABI)

teecall\_mem\_reclaim

Recall that `page_map_remove` operation puts the removed page and the hcat mapping structure pages in a Confidential-Unassigned state.

I.e this ensure no TVM mappings can be active for these pages

1. VMM may reclaim only pages that are in Confidential-Unassigned state
2. The reclaim TSM operation:
  - a. Verifies that the PAs referenced are either Non-confidential (No-operation) or Confidential-Unassigned state
  - b. TSM takes exclusive lock over the MTT tracker entry for the PA
    - i. TSM (lazily) scrubs page contents
    - ii. TSM updates MTT tracker entry (synchronized) for the page as Non-confidential
  - c. TSM returns the PA as an Non-Conf page to the VMM
  - d. VMM translations to the PA (via 1st or G stage mappings) may be created now

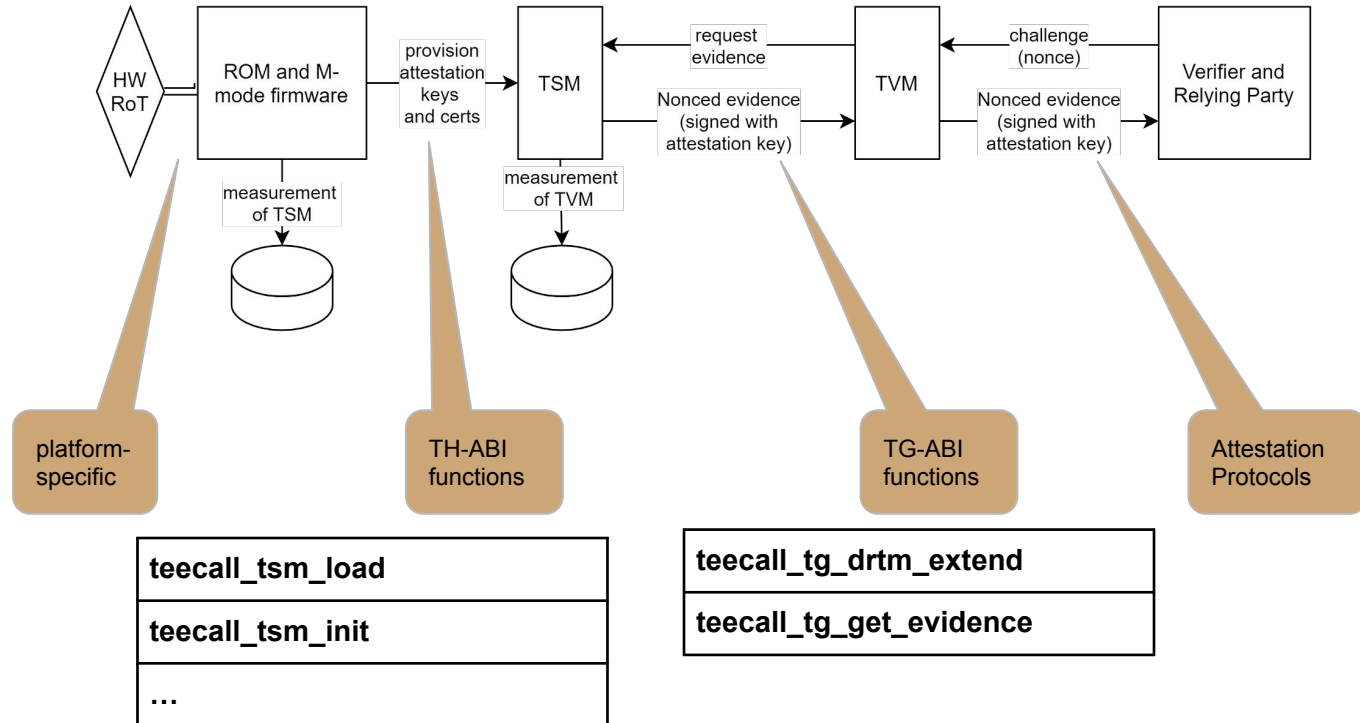


# Extra Slides

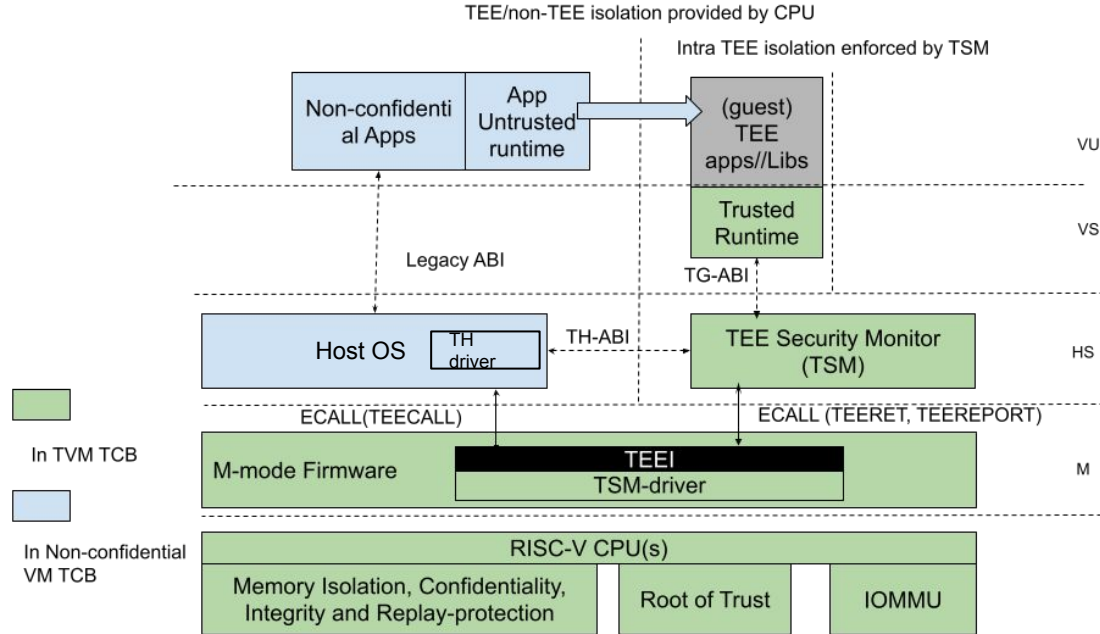
# TG spec dev/discussion areas

- SW Lifecycle
- Confidential Memory Mgmt
- Attestation
- Physical Protection - Encryption
- Interrupt Mgmt
- Direct IO assignment to TVMs
  - IOMMU discussions ongoing to establish direction
- Confidential process (SW model)
- Domain partitioning
- Stolen time
- TVM Live Migration
- TVM Checkpointing
- Others?

# TVM Attestation



# Ref. Arch use for confidential applications



# Role of M-mode

- M-mode firmware hosts the TSM-driver that programs HW to provide the isolation of the TSM memory via following mechanisms:
  - Cryptographic mechanisms (confidentiality, integrity) and/or
  - Access-control mechanisms (page ownership tracking or sequestered memory)
  - Owns context switch from host OS/VMM to TSM and vice versa (TEE ABI functions invoked via ECALL to M-mode (TEECALL, TEERET, TEEREPOROT)
    - Activates “AP-TEE mode” on the hart
- M-mode programs HW engines for access-control, memory confidentiality, and RoT for TSM reporting.

# Role of the TSM

TSM enforces isolation of TVM memory (amongst TVMs) via 2nd stage translation

TSM enforces isolation of TVM hart state when invoked by the host VMM to schedule TVM virtual-hart on a physical hart.

TSM maintains TVM measurement for attestation

TSM itself measured by the TSM-driver and HW RoT.