Hardening and Adapting Trusted Execution Environments for Emerging Platforms

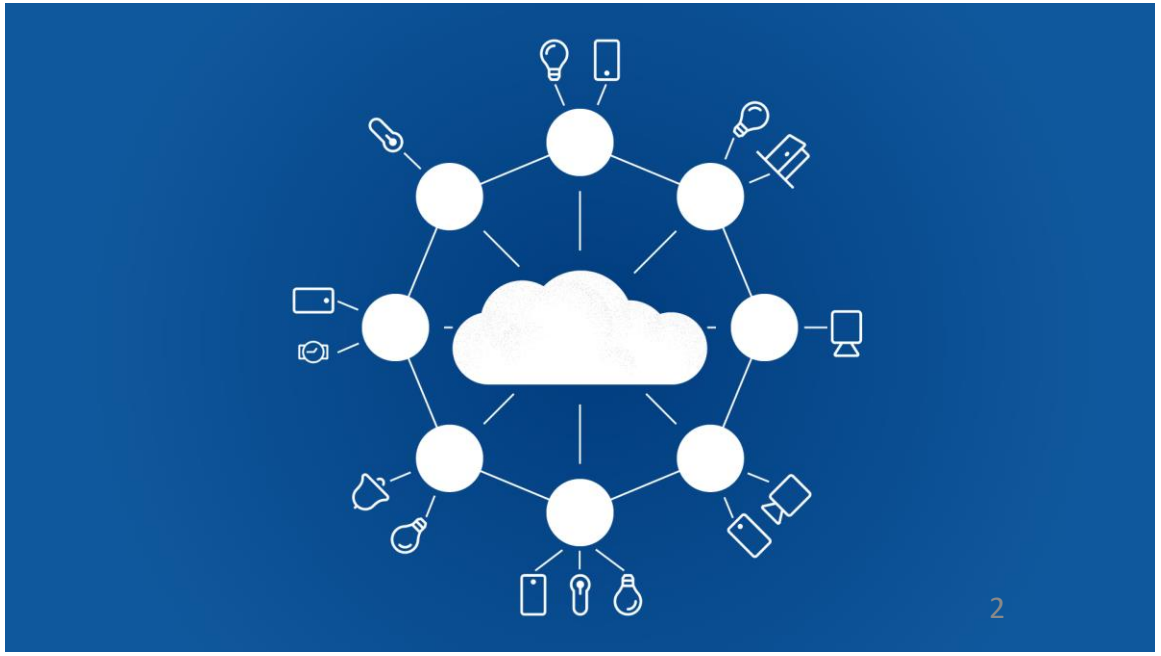
Ph.D. Defense of Dissertation

Fan Sang

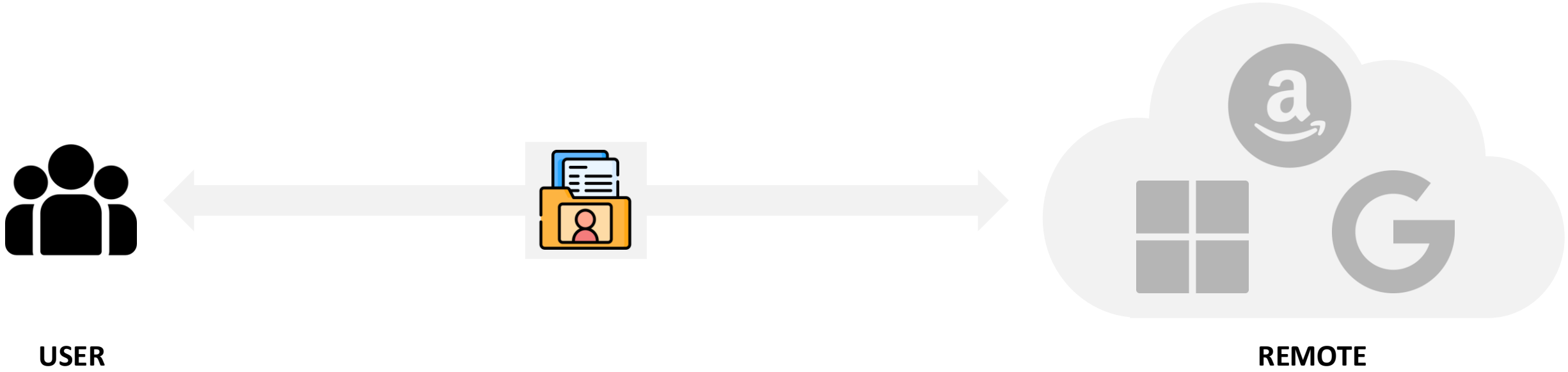
Advisor: Prof. Taesoo Kim

Georgia Tech College of Computing
School of Cybersecurity and Privacy

Prevalence of Data



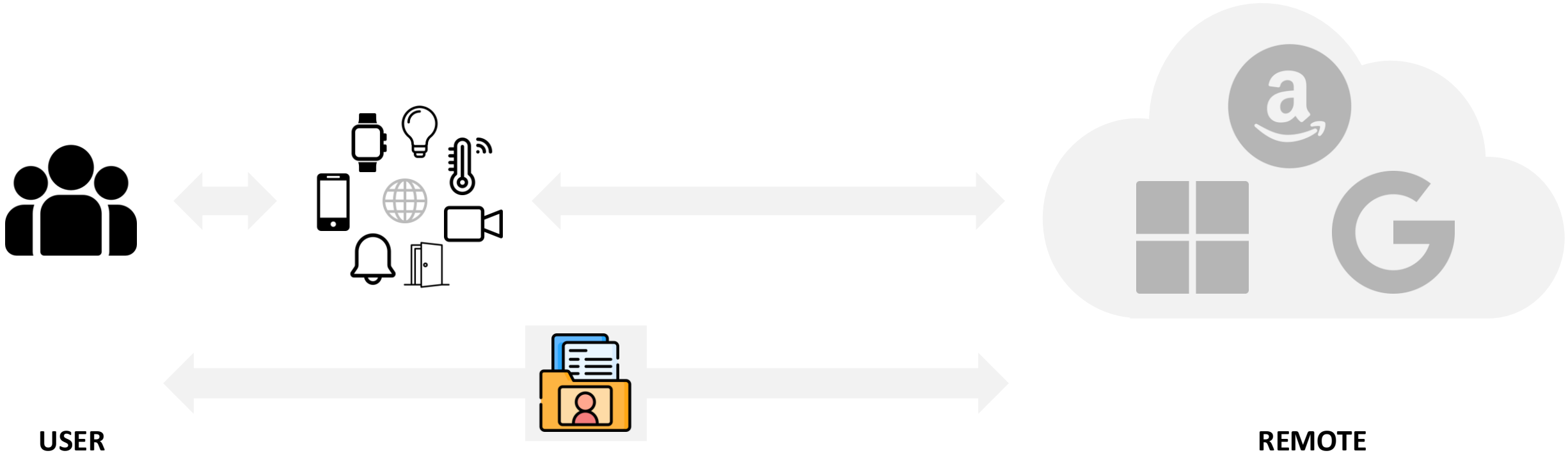
Prevalence of Data – Cloud Computing



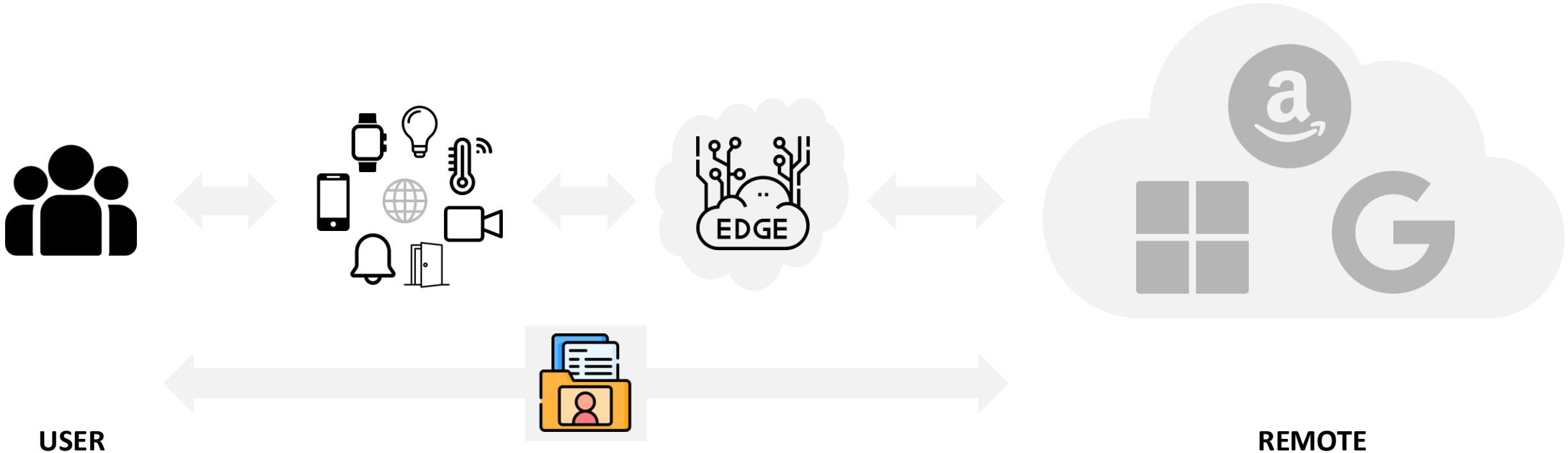
USER

REMOTE

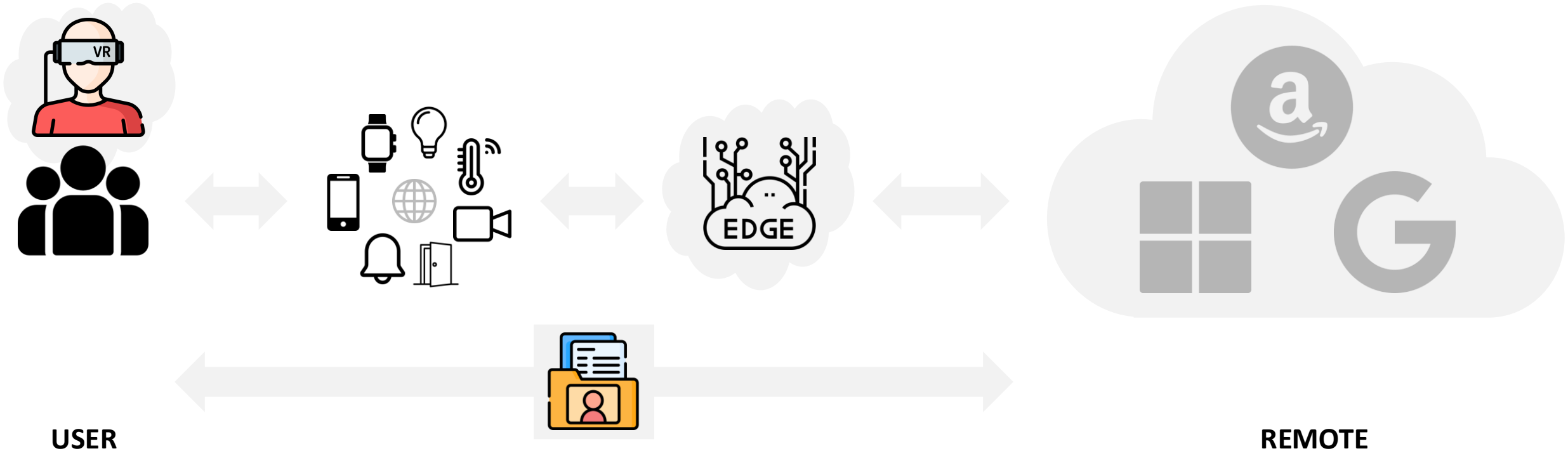
Prevalence of Data – IoT



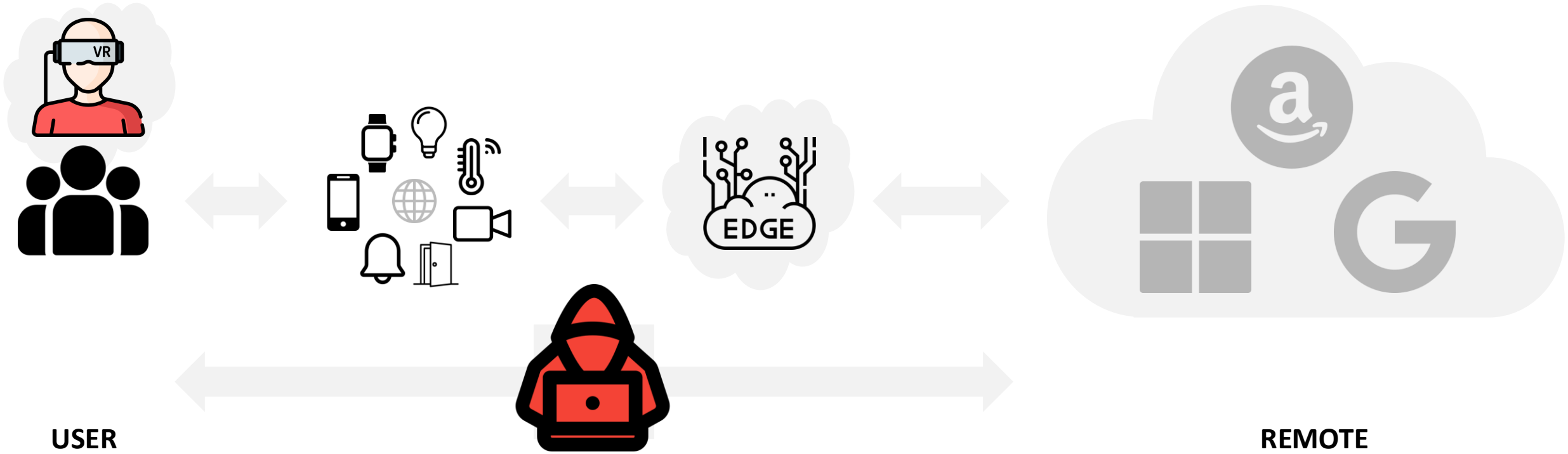
Prevalence of Data – Edge Computing



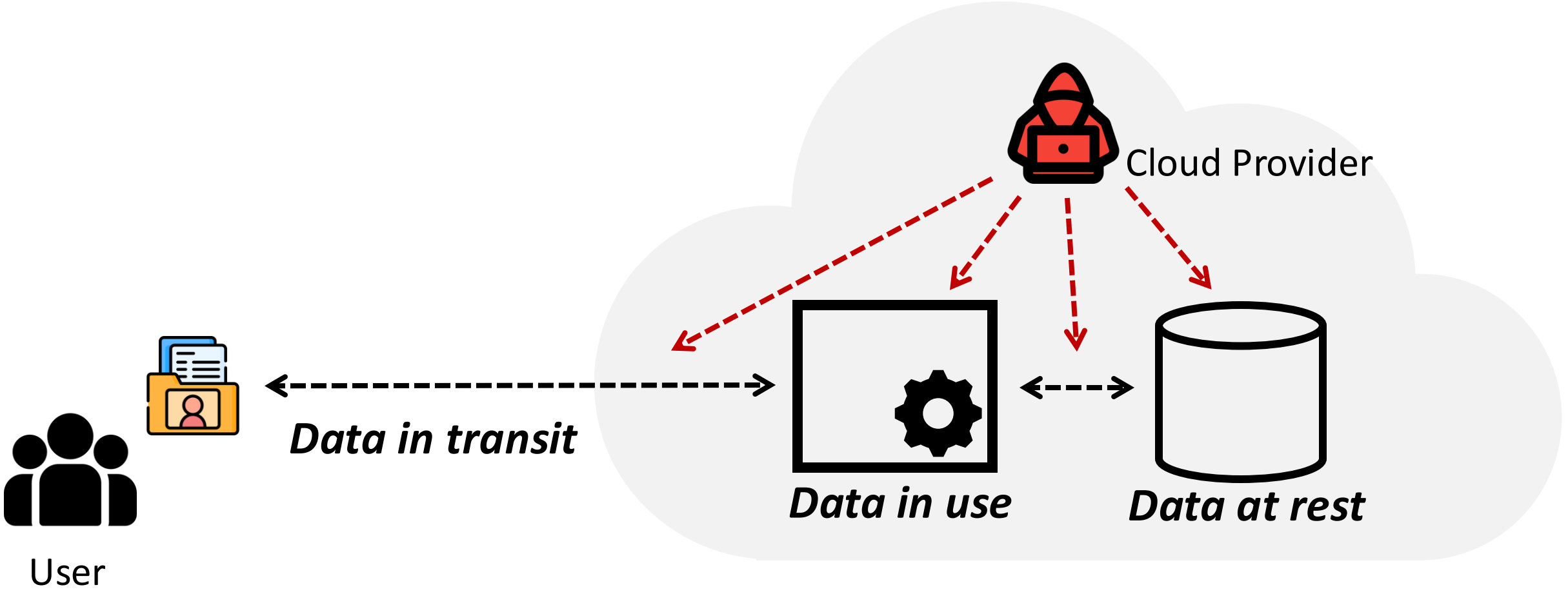
Prevalence of Data – Emerging Platforms (VR)



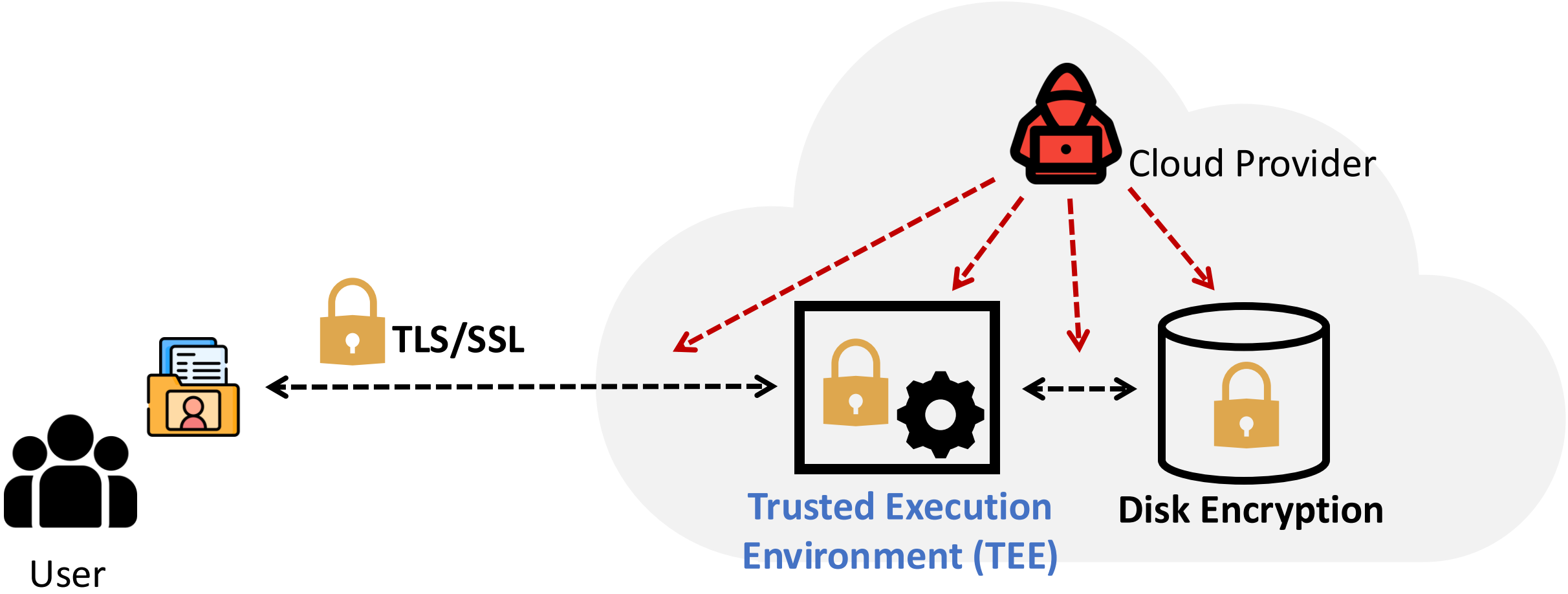
Data Security – Privileged Attackers



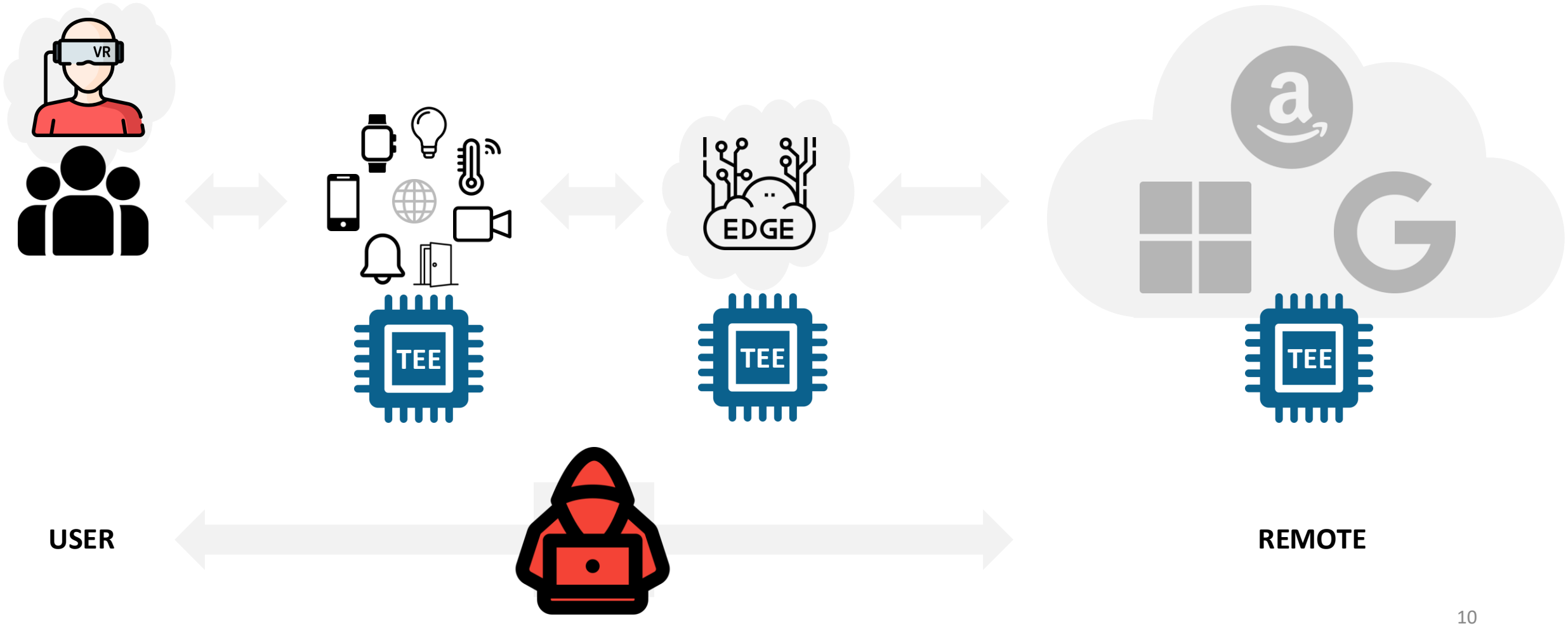
Data Security – Cloud as an Example



Existing Security Solutions

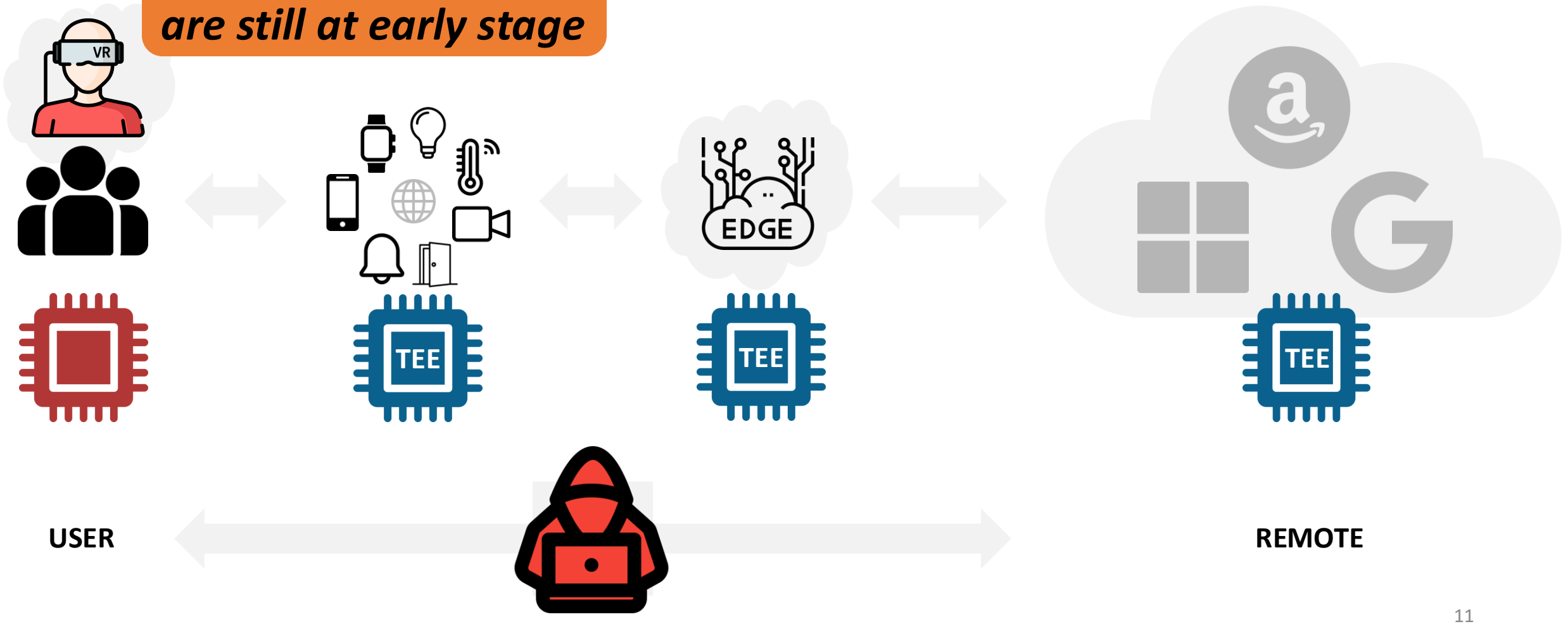


Prevalence of TEEs



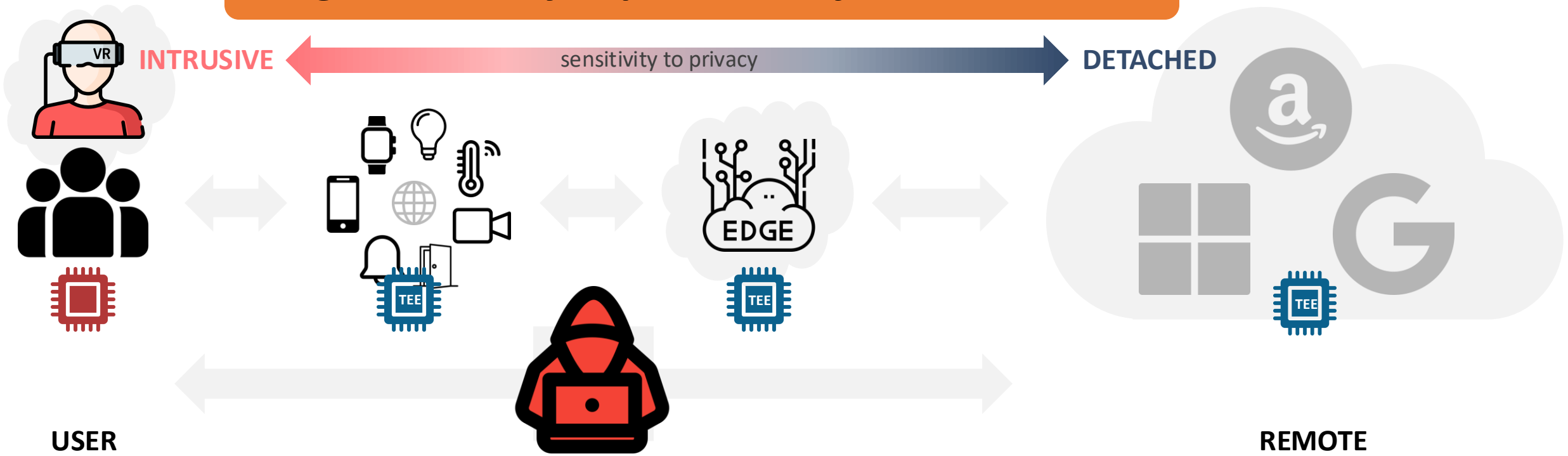
Prevalence of TEEs – Emerging Platforms

Emerging platforms are still at early stage



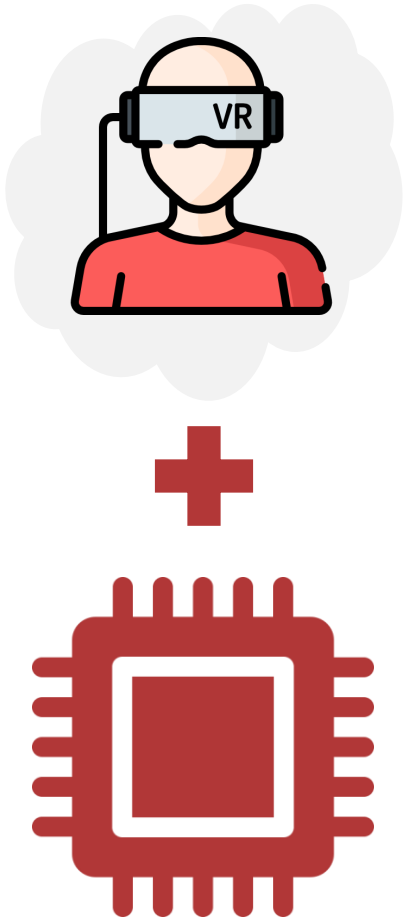
Necessity of Adapting TEEs for VR Platforms

Higher security expectations from end users



New regulations regarding consumer privacy

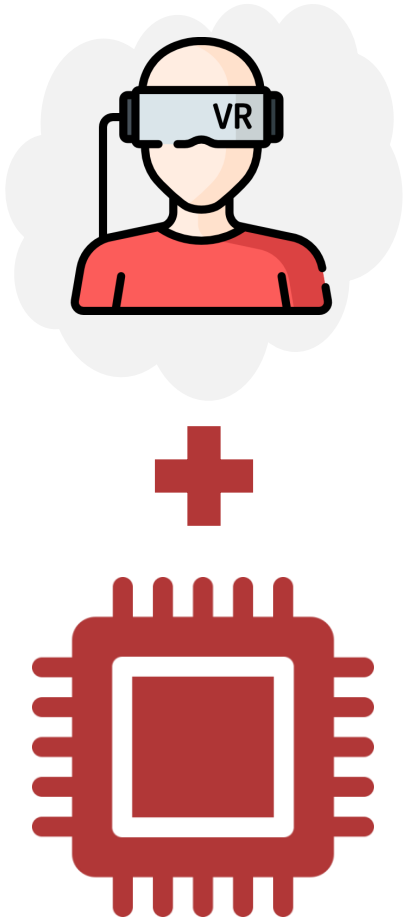
Adapt TEEs for VR Platforms



Higher security expectations from end users

New regulations regarding consumer privacy

Adapt TEEs for VR Platforms



Security Issues of Current TEEs

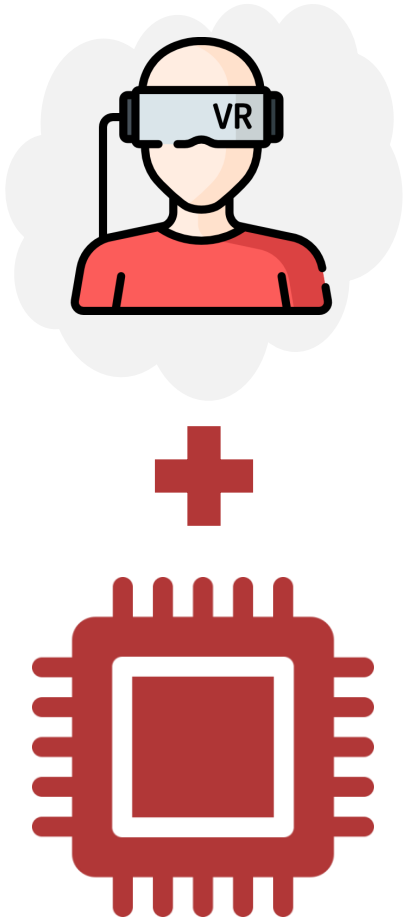
Limited Knowledge about Security of VR

New Requirements to Satisfy during Adaption

Higher security expectations from end users

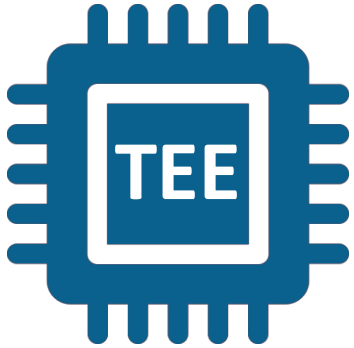
New regulations regarding consumer privacy

Adapt (Secure) TEEs for (Insecure) VR Platforms



Security Issues of Current TEEs	**Harden Current TEEs**
Limited Knowledge about Security of VR	**Understand Security of VR**
New Requirements to Satisfy during Adaption	**Adapt with Requirements**

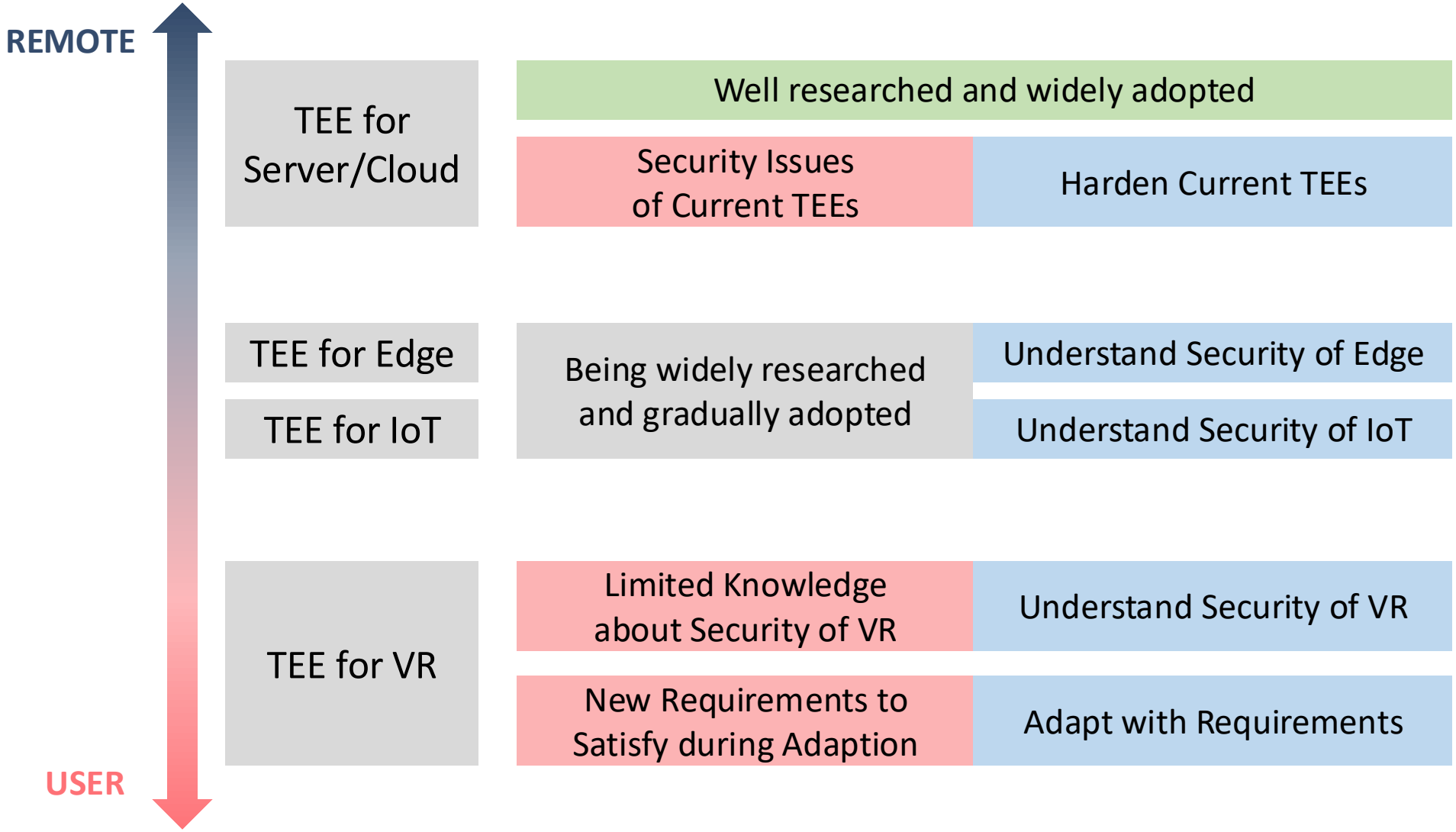
Adapt (Secure) TEEs for (Insecure) VR Platforms



Security Issues of Current TEEs	**Harden Current TEEs**
Limited Knowledge about Security of VR	**Understand Security of VR**
New Requirements to Satisfy during Adaption	**Adapt with Requirements**

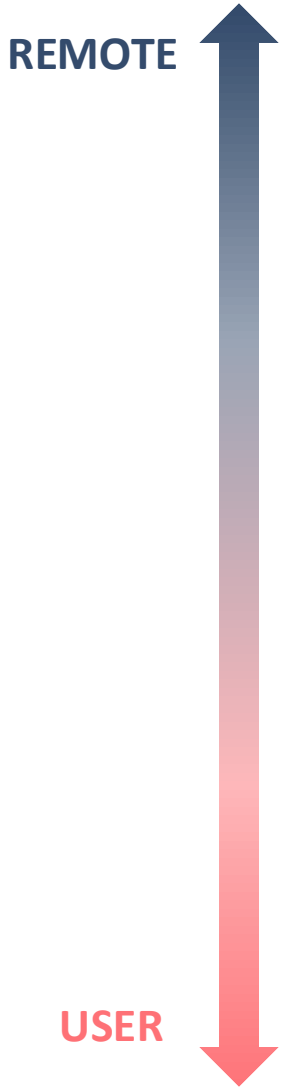
Ph.D. Research

Harden TEEs
Understand Security
Adapt TEEs



Ph.D. Research

Harden TEEs
Understand Security
Adapt TEEs



TEE for Server/Cloud

Harden Current TEEs

- **Pridwen** [ATC'22]: Defending multiple SCAs simultaneously on diverse platforms
- **Sense** [NDSS'24]: Tackling SCAs from the hardware root cause

TEE for Edge

Understand Security of Edge

- **Graphene** [ACSAC'24 in submission]: Threat analysis of edge systems via attack graphs

TEE for IoT

Understand Security of IoT

- **Rome** [TMC in submission]: Password-enabled role-based smart home access control

TEE for VR

Understand Security of VR

- **SNAP** [CCS'21]: Hardware support to improve fuzzing performance and precision
- **Mirage** [Eurosys'25 in submission]: Understanding security issues in VR via fuzzing

Adapt with Requirements

- **Rakis** [Eurosys'25 in submission]: Certifying untrusted fast I/O primitives for TEEs
- **Portal** [Oakland'25 in submission]: Adapt Mobile TEEs for VR platforms

Ph.D. Research

Harden TEEs
Understand Security
Adapt TEEs

REMOTE

USER

TEE for Server/Cloud

Harden Current TEEs

- **Pridwen** [ATC'22]: Defending multiple SCAs simultaneously on **diverse platforms**
- **Sense** [NDSS'24]: Tackling SCAs from the **hardware root cause**

TEE for Edge

Understand Security of Edge

- **Graphene** [ACSAC'24 in submission]: Threat analysis of edge systems via attack graphs

TEE for IoT

Understand Security of IoT

- **Rome** [TMC in submission]: Password-enabled role-based smart home access control

TEE for VR

Understand Security of VR

- **SNAP** [CCS'21]: Hardware support to improve fuzzing performance and precision
- **Mirage** [Eurosys'25 in submission]: Understanding security issues in VR via fuzzing

Adapt with Requirements

- **Rakis** [Eurosys'25 in submission]: Certifying untrusted fast I/O primitives for TEEs
- **Portal** [Oakland'25 in submission]: Adapt Mobile TEEs for VR platforms

Ph.D. Research

Harden TEEs
Understand Security
Adapt TEEs

REMOTE

TEE for Server/Cloud

Harden Current TEEs

- **Pridwen** [ATC'22]: Defending multiple SCAs simultaneously on **diverse platforms**
- **Sense** [NDSS'24]: Tackling SCAs from the **hardware root cause**

- **Pridwen: Passively harden *unknown platforms* with multiple defenses**
- **Sense: Proactively harden *microarchitecture* with hardware extension**

→ With emerging platforms in mind

TEE for VR

Understand Security of VR

- **SNAP** [CCS'21]: Hardware support to improve fuzzing performance and precision
- **Mirage** [Eurosys'25 in submission]: Understanding security issues in VR via fuzzing

Adapt with Requirements

- **Rakis** [Eurosys'25 in submission]: Certifying untrusted fast I/O primitives for TEEs
- **Portal** [Oakland'25 in submission]: Adapt Mobile TEEs for VR platforms

USER

Ph.D. Research

Harden TEEs
Understand Security
Adapt TEEs

REMOTE

USER

TEE for Server/Cloud

Harden Current TEEs

- **Pridwen** [ATC'22]: Defending multiple SCAs simultaneously on diverse platforms
- **Sense** [NDSS'24]: Tackling SCAs from the hardware root cause

TEE for Edge

Understand Security of Edge

- **Graphene** [ACSAC'24 in submission]: Threat analysis of edge systems via attack graphs

TEE for IoT

Understand Security of IoT

- **Rome** [TMC in submission]: Password-enabled role-based smart home access control

TEE for VR

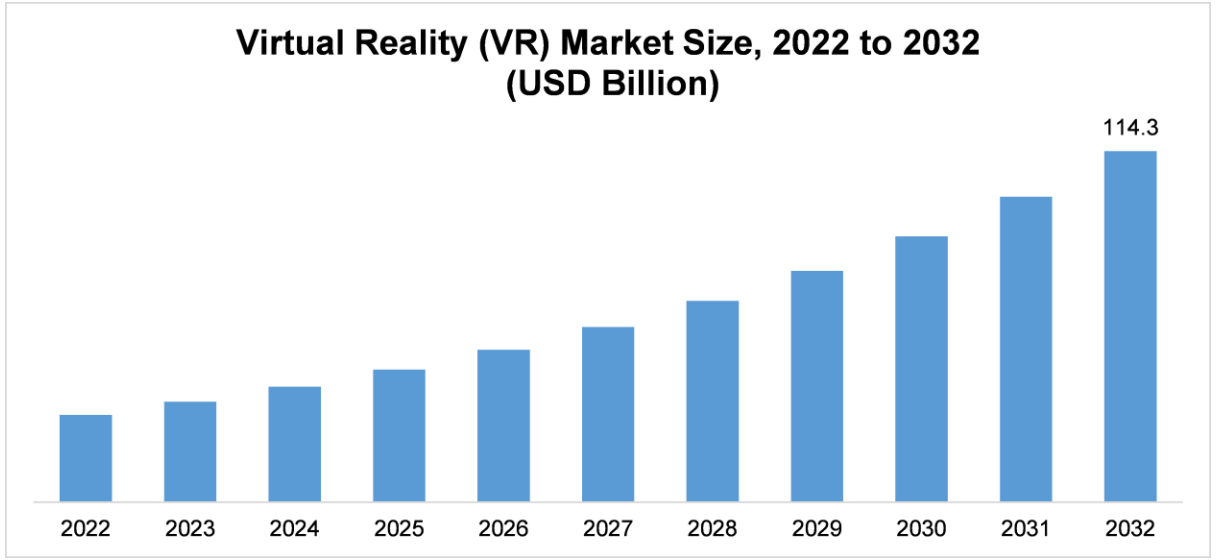
Understand Security of VR

- **SNAP** [CCS'21]: Hardware support to improve fuzzing performance and precision
- **Mirage [Eurosys'25 in submission]: Understanding security issues in VR via fuzzing**

Adapt with Requirements

- **Rakis** [Eurosys'25 in submission]: Certifying untrusted fast I/O primitives for TEEs
- **Portal [Oakland'25 in submission]: Adapt Mobile TEEs for VR platforms**

Virtual Reality (VR) – A Fast-Emerging Platform



Virtual Reality (VR) – Testing

Meta

"During my internship at Meta, each intern was asked to participate in a 1-hour VR session every week to help test the product."

--- Anonymous 1

"Meta has hundreds of physical VR devices running in a warehouse everyday to test the product."

--- Anonymous 2

Challenges of Testing VR Applications (#1)

Research

Code-level

Open source PoC

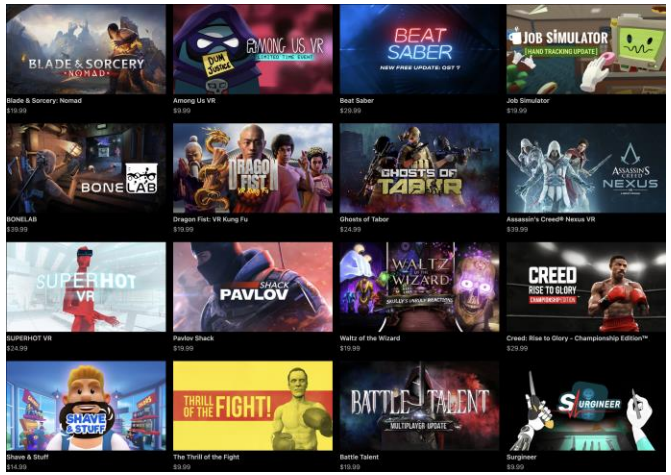
Lack complexity

Inactive

Commercial

Proprietary

Complex and large



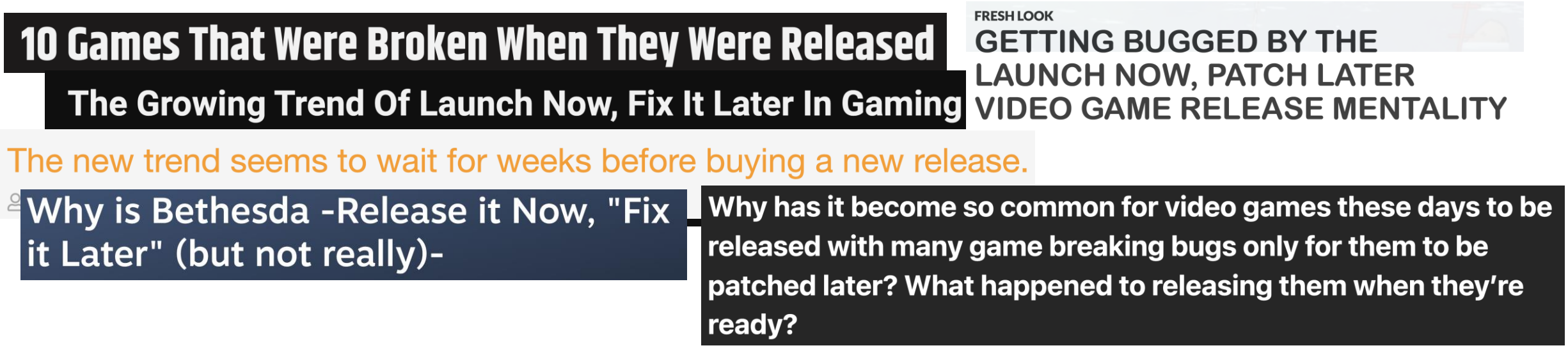
Vender application store

→ Vendors must verify if applications meet claimed functionalities and adhere to platform policies through black-box testing.

Challenges of Testing VR Applications (#2)

Issues at the initial release, despite testing efforts.

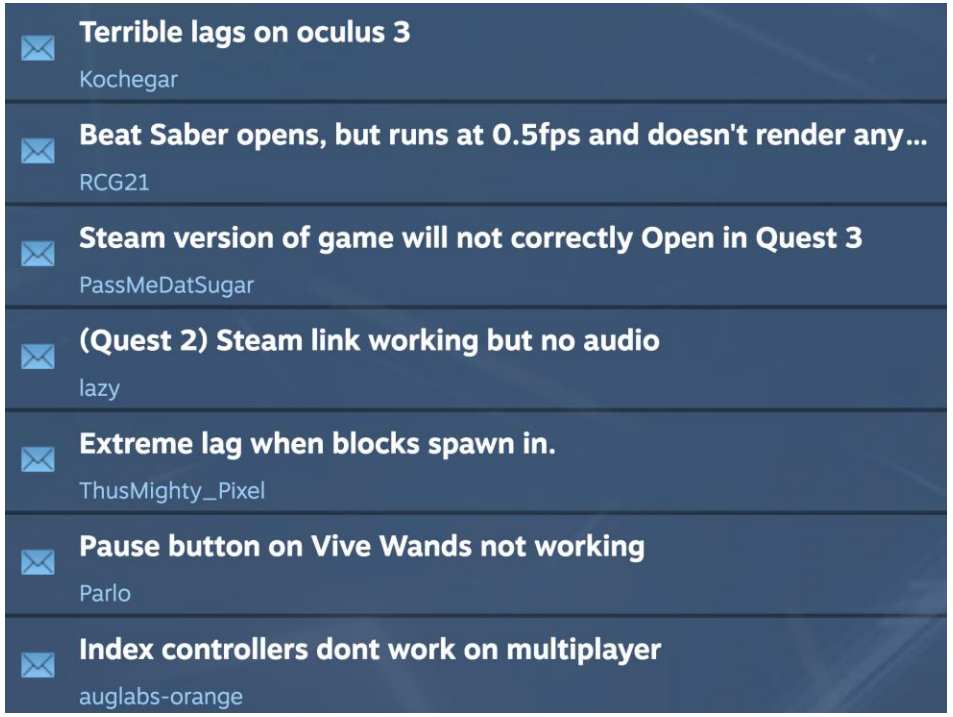
- **Huge input space** of **realistic** user and system behaviors post-release



→ Code-level testing cannot capture such issues due to the lack of realistic environment.

Challenges of Testing VR Applications (#3)

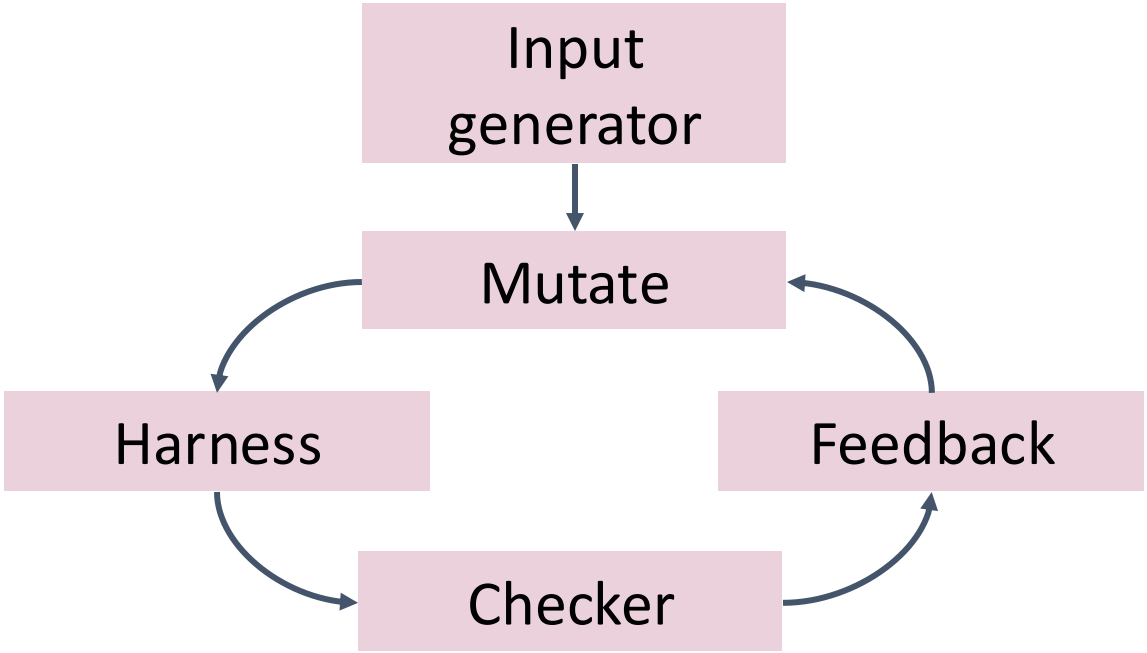
- **Complains most about QoE**
  - Not traditional software issues
- Priority of VR vendors
  - **Quality of immersiveness**
- Existing testing focuses on functionality
  - **Overlooks VR QoE issues**



→ The effectiveness of code-level testing is limited in the commercial VR ecosystem that demands high-quality experiences

Fuzzing

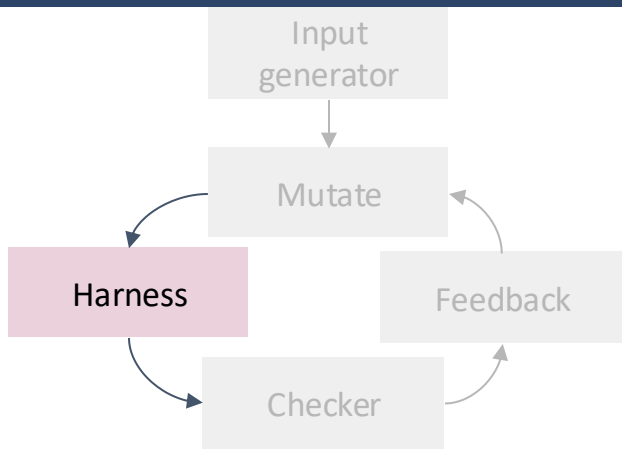
- VR is **highly closed**, yet exists **unique challenges in testing**
- **Fuzzing** helps find issues in an enormous input space more efficiently
  - Favoring inputs that are more likely to trigger defined issues



Mirage: Black-box Testing for COTS VR Apps

Testing without Physical Devices

- Harness executes the inputs on the target system
- In-house virtual VR driver for **automation**
- The virtual driver fully **simulates** a real device
  - Control inputs
  - Properties



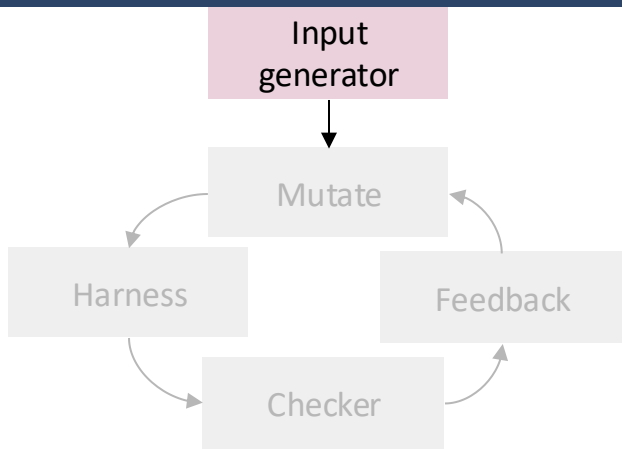
STEAMVR

How it interacts with apps?

Which specific device?

Input: Device Properties

- Int, long, bool, float, string, matrix types
- ~100 properties for a specific device
- Represents a real device by the driver to the VR runtime

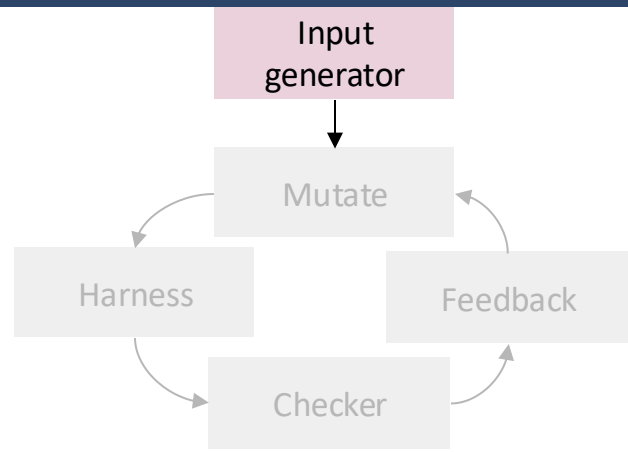


Which specific device?

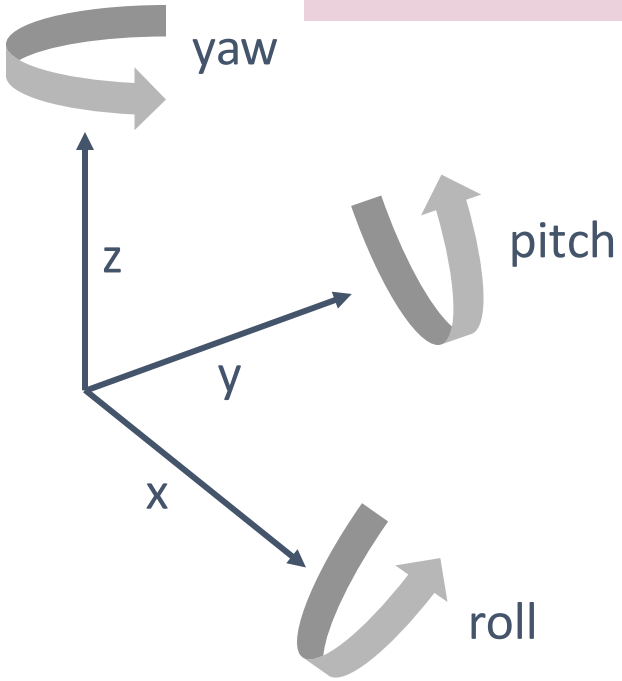
- Model
- Serial #
- Firmware version
- Has camera
- Display related
- Supported buttons
- etc.

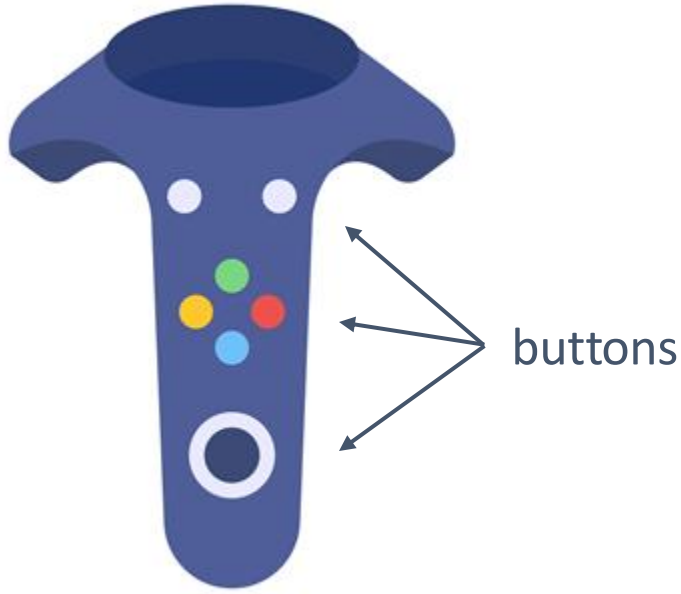
Input: Device Pose

- Rotation
  - Yaw, pitch, roll
- Position
  - <X, Y, Z>
- Buttons
  - Menu
  - Grip
  - System
  - Trackpad



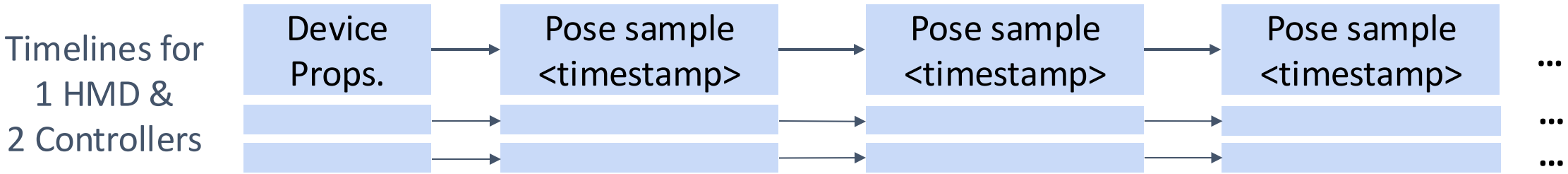
How it interacts with apps?





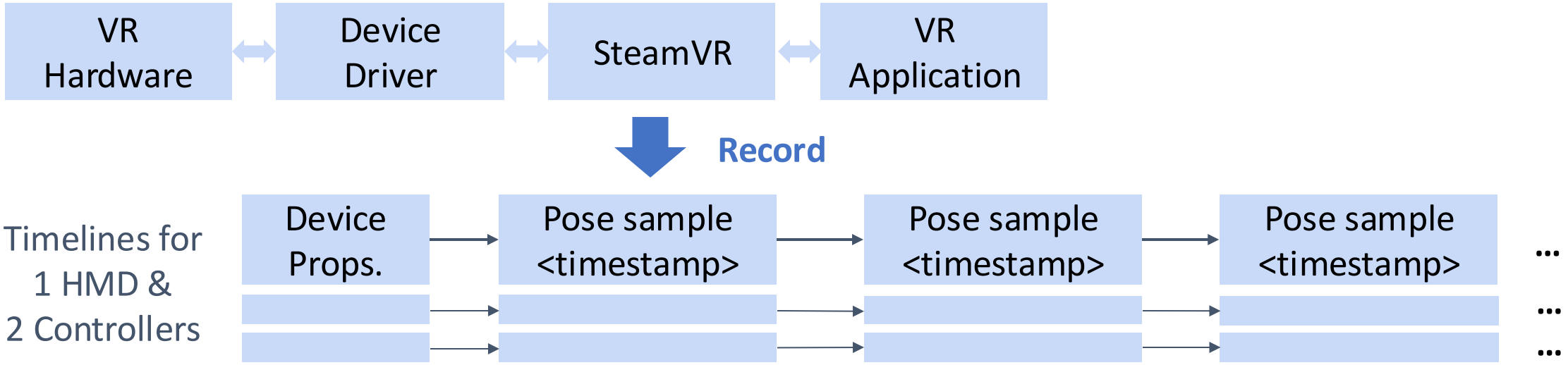
Input Format

- **Time series** of **<pose>** for a device with **<properties>**



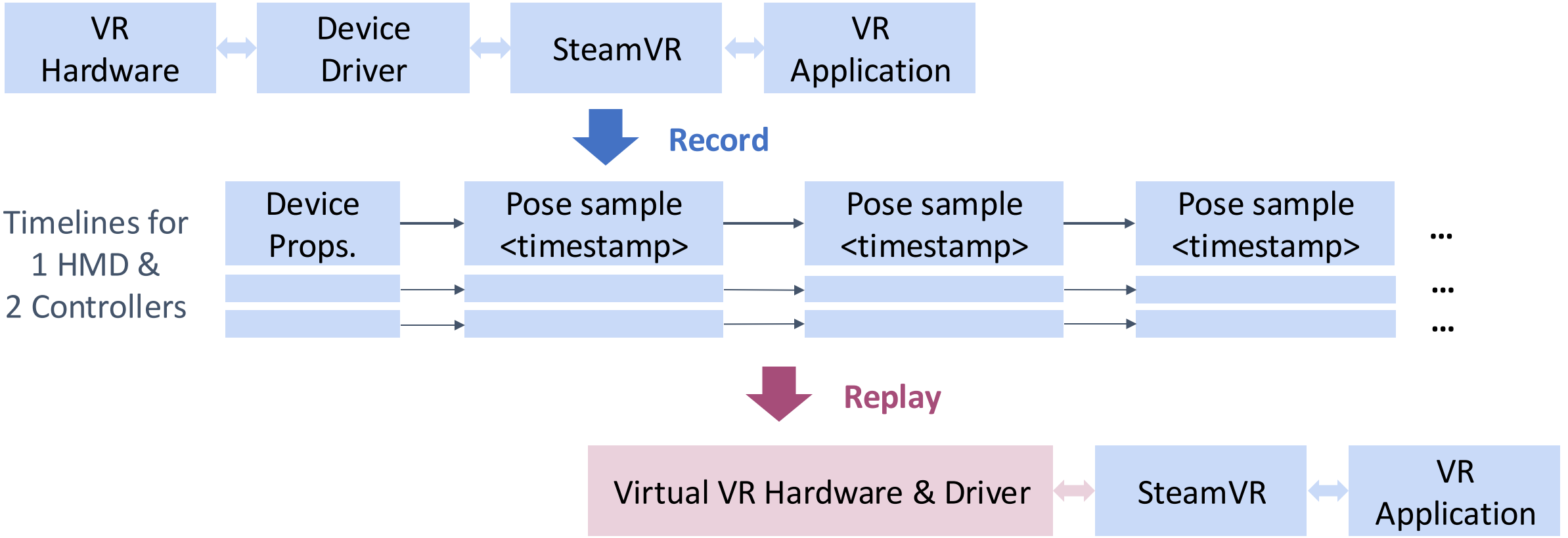
Seed Input Using Physical Devices

- Manually constructing human inputs for an app is hard
- Record interaction in physical devices as seed inputs



Harness Execution

- Virtual devices interacts with apps by replaying recording
- Replay the (mutated) inputs during fuzzing

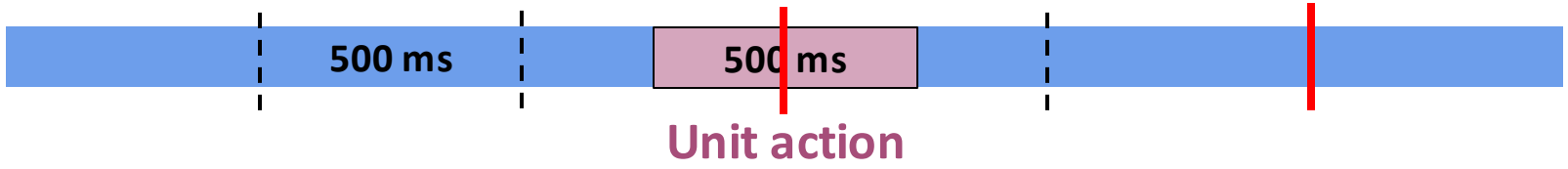


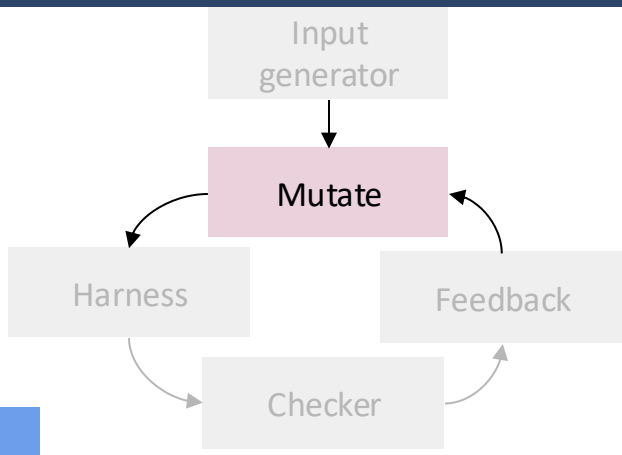
Preparation Recording

- VR applications typically begin with menu interfaces or setup pages
  - **Repeated each time** the application starts before accessing the main scenes
- Preparation is essential for automated VR testing
  - Necessary user interactions to reach the main testing scene
  - Main testing scene involves critical application logic and complex interactions
- Preparation recording
  - Covers all initial and final interactions needed for cycling the testing
  - *Executed before each test input*
  - *Excluded from mutation*

Mutation

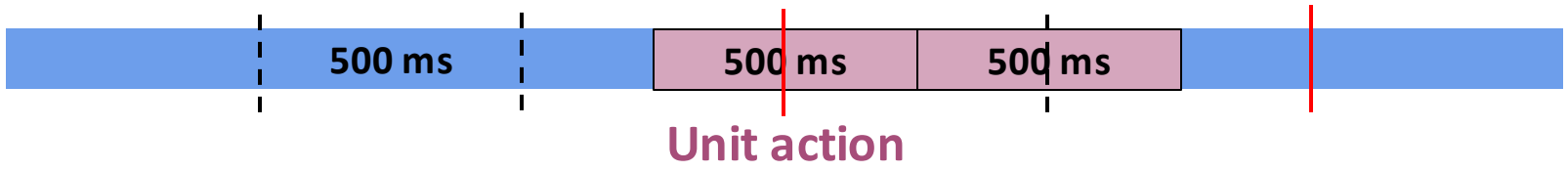
- 500 ms input period as a unit action to mutate
  - Insert, delete, modify, replace





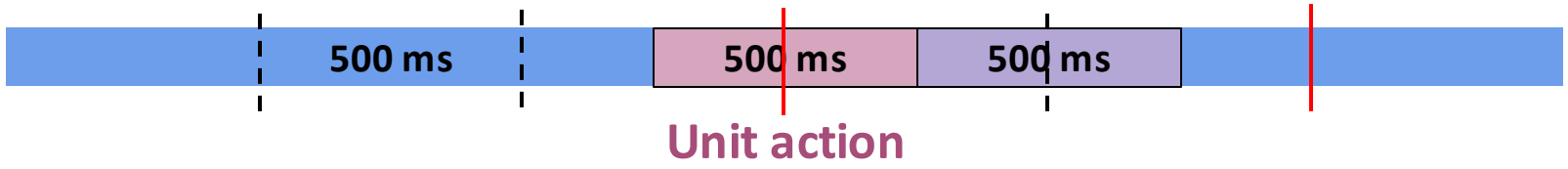
Insertion

- Insert by duplicating before & after
  - I.e., repeat the action



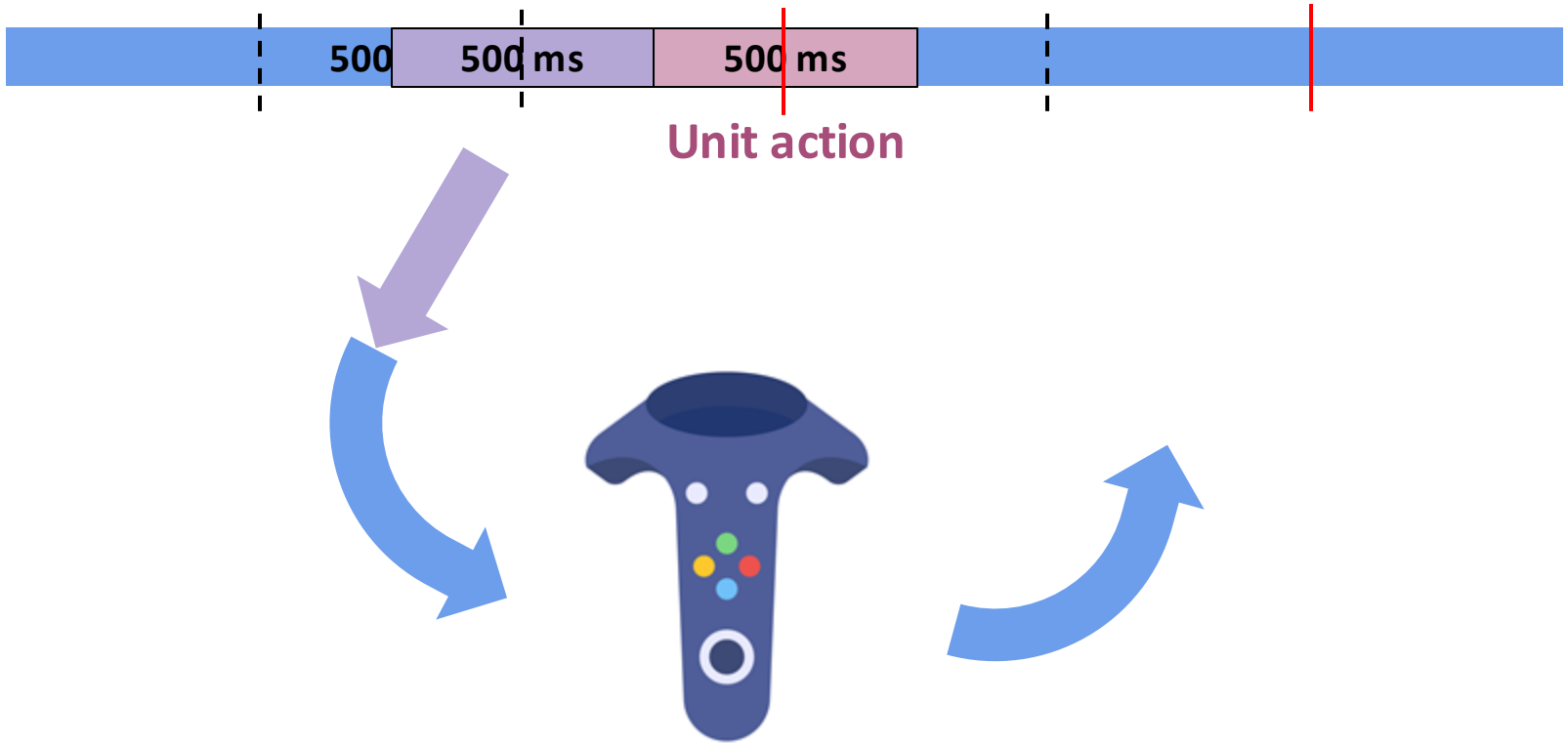
Insertion

- Insert by extending after
  - Extend along tail (i.e., exaggerate the interesting movement)



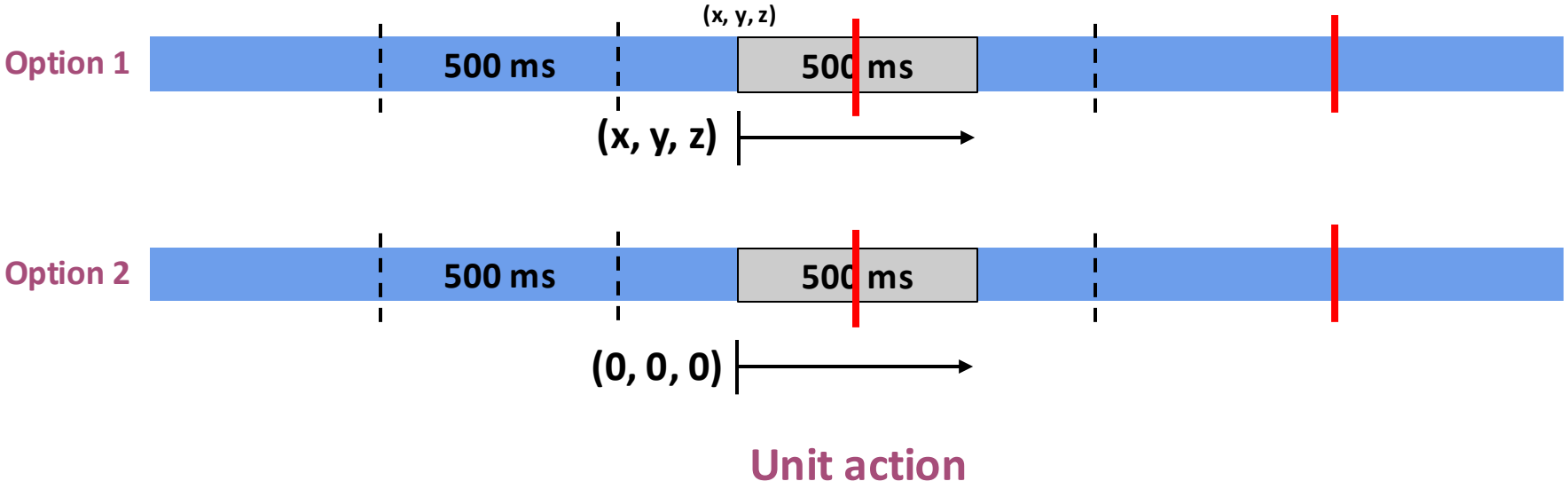
Insertion

- Insert by extending before
  - Extend along head (i.e., exaggerate the interesting movement)



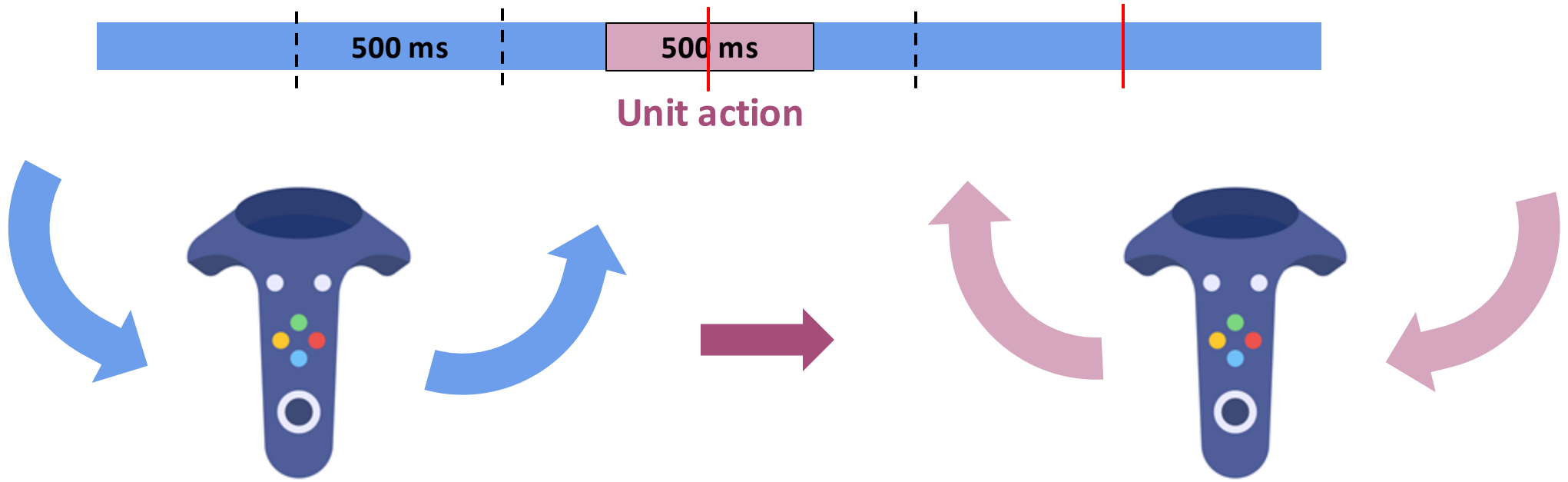
Deletion

- Delete the target unit action
  - Option 1: stop and stay
  - Option 2: stay at the origin (0, 0, 0)



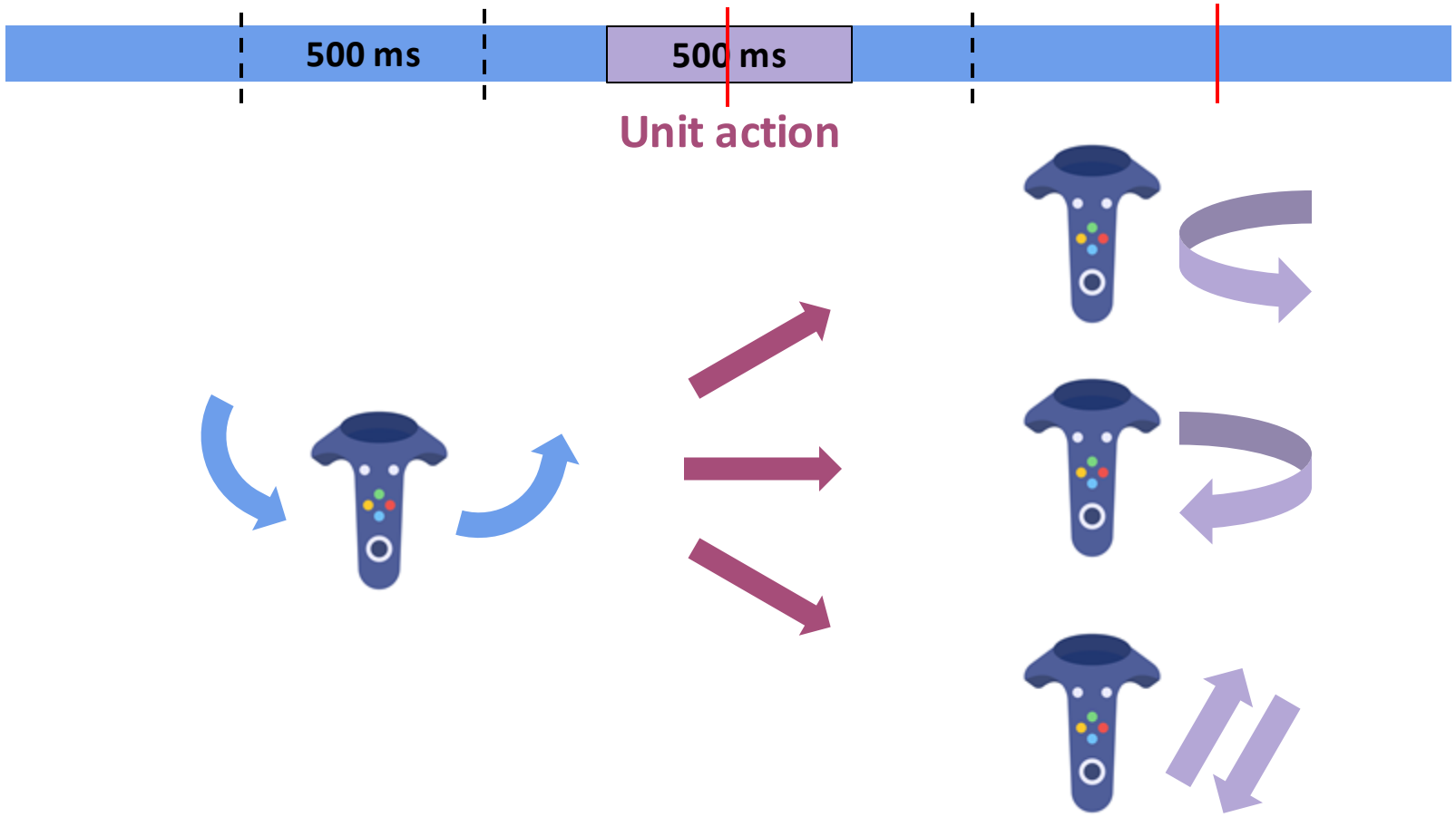
Modification

- Modify in-place
  - Increased/decreased speed
  - Reverse direction
  - Edge values



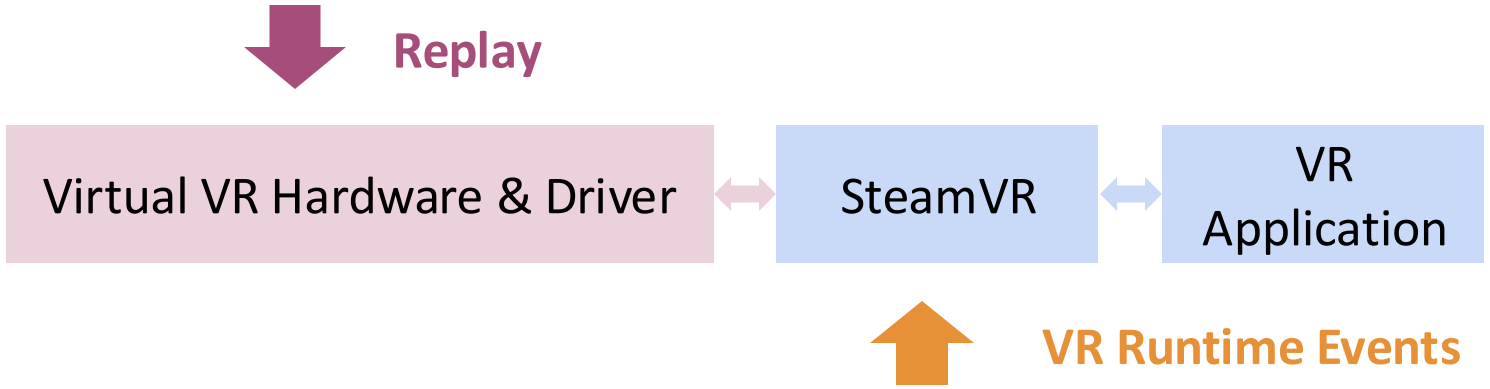
Modification

- Replace with pre-defined unit actions
  - e.g., swing, circle, shake, etc.



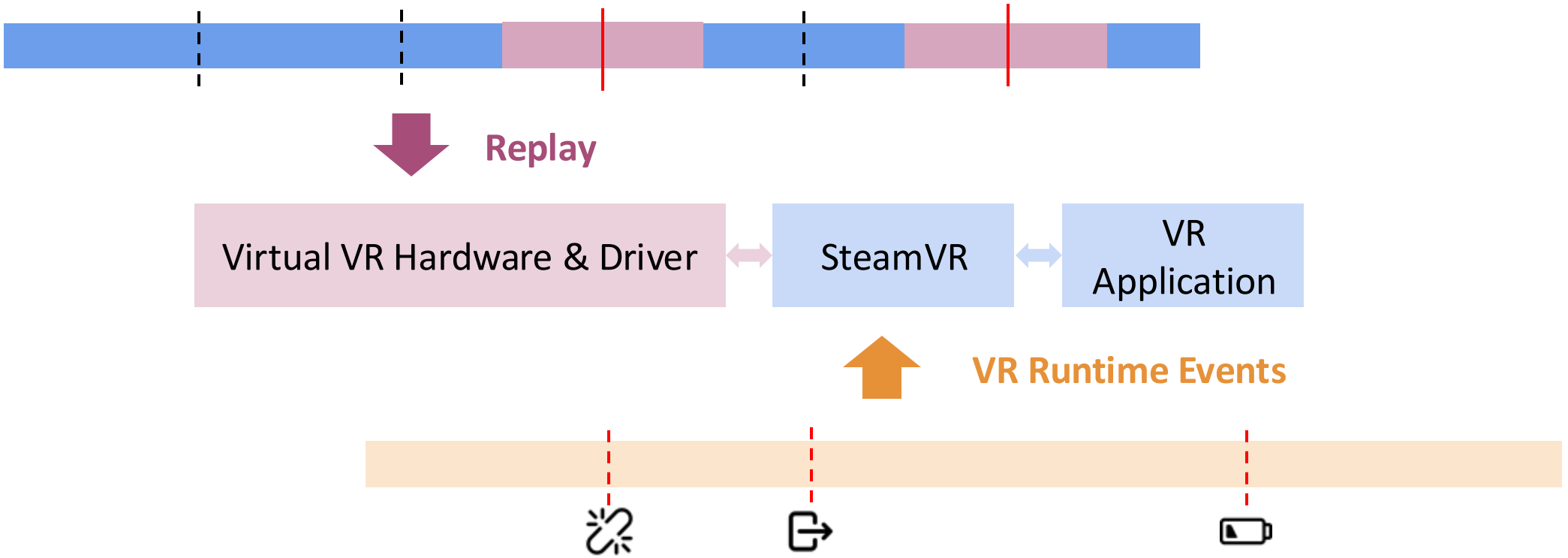
Other Mutation Events

- We can inject non input-related events during the replay
- Inject VR runtime events
  - Device disconnect, reconnect, standby
  - Device low battery
  - Play area trespass warning
  - Etc.



Other Mutation Events

- An individual VR runtime event time series
  - [type] of event to insert at [timestamp]
- 2-D compositional input

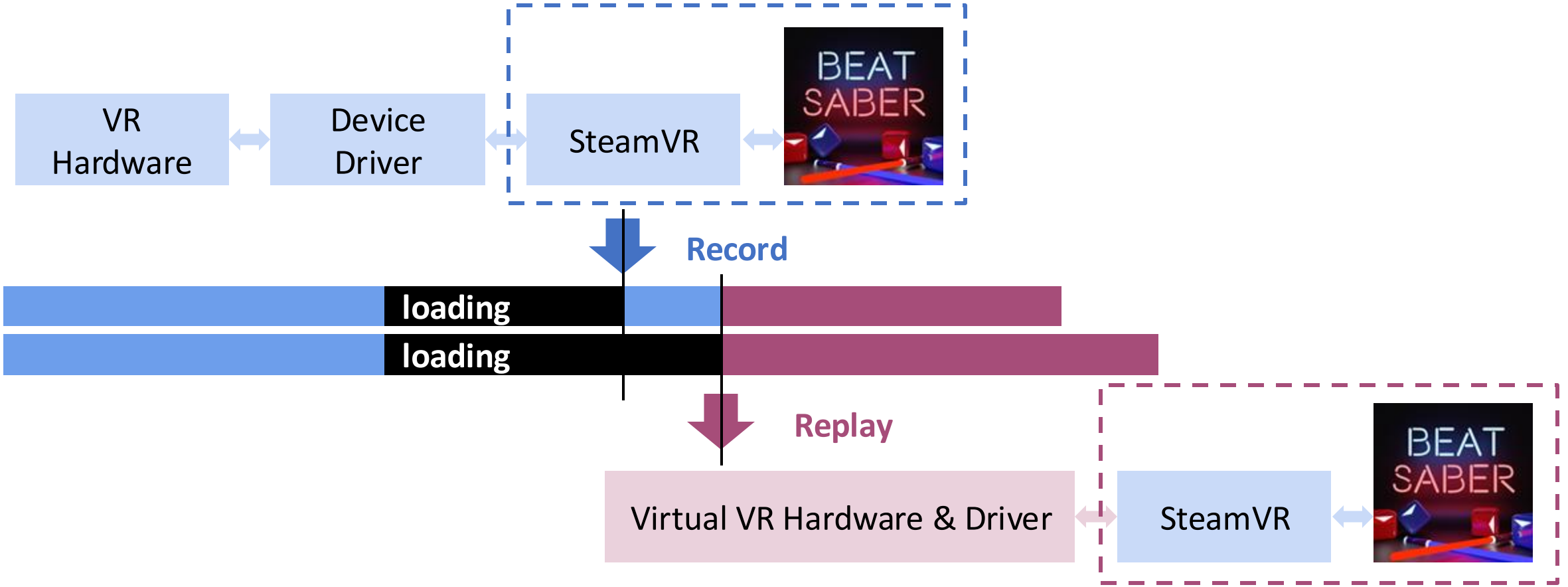


Challenge: Reliability of Replay

- Replay relies **on time sensitive operations**
- Observation: The movements are slightly misaligned every time replayed
  - Affect the correct execution of replays
- Root cause: non-deterministic loading time

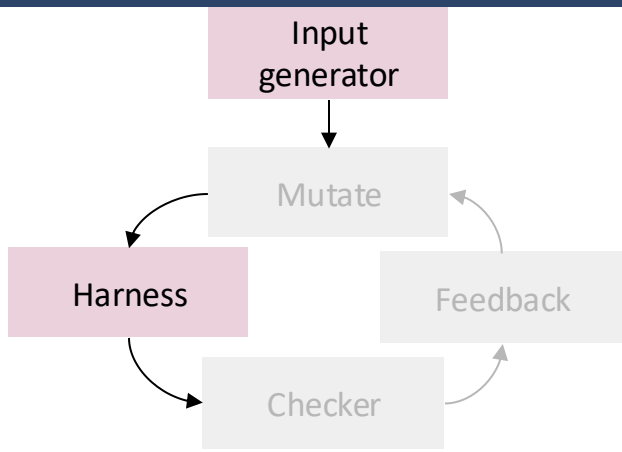
Reliability of Replay

- The app states are different during recording and replay
- Non-deterministic loading mis-aligns the time series



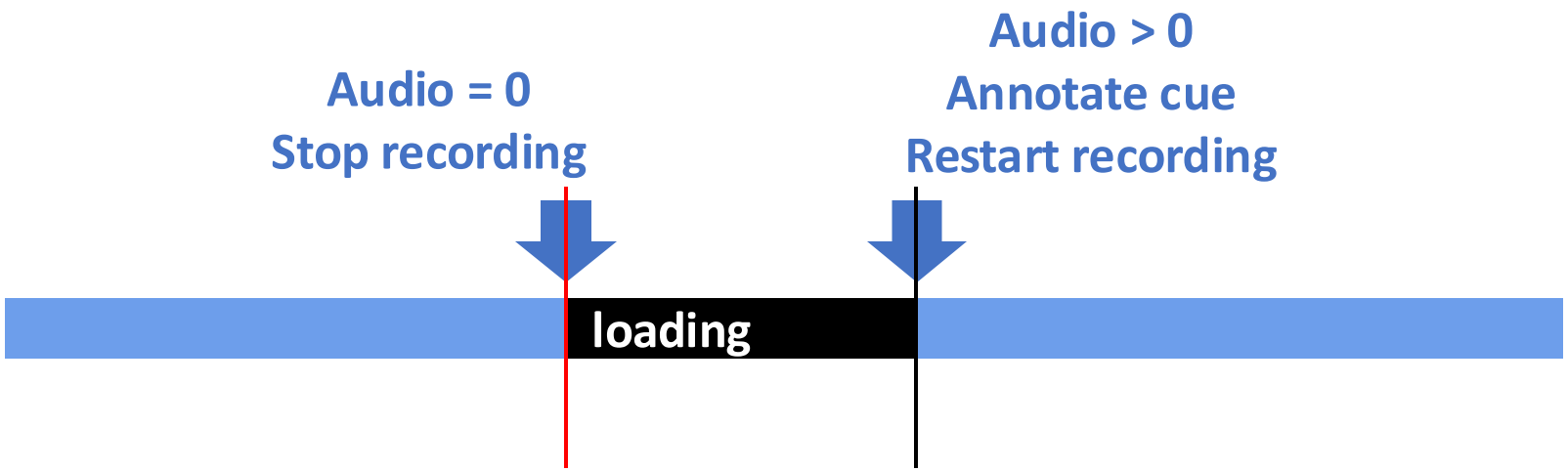
Solution – Synchronization Cues

- Observation: No audio while loading (audio == 0)
- Audio Cue to sync record and replay



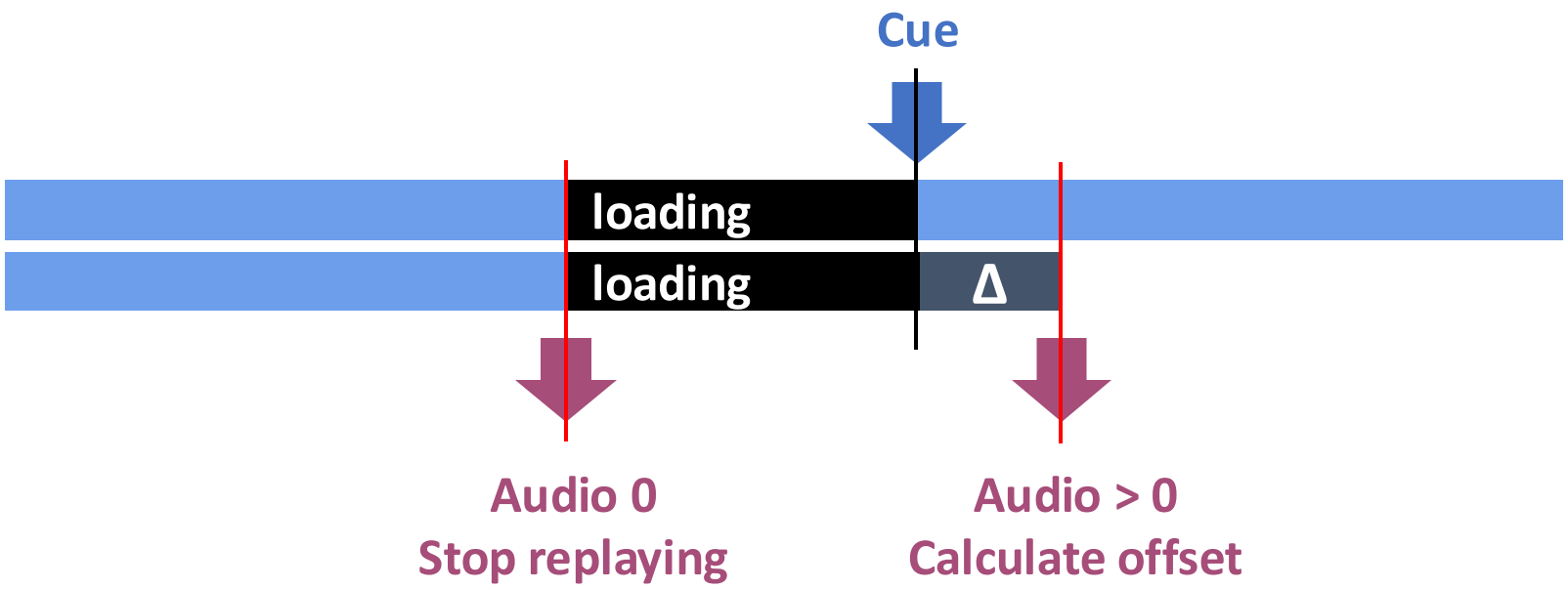
Synchronization Cues

- Audio Cue to sync record and replay
  - Annotate the cue when loading finishes during recording



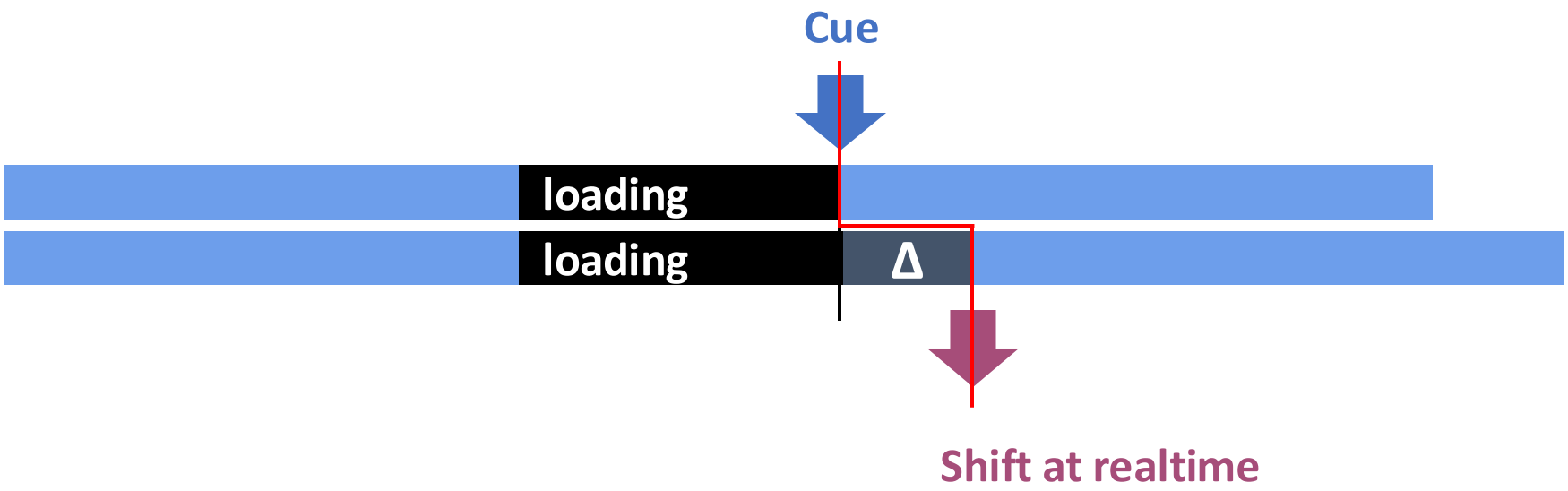
Audio Cues

- Correctly calculate audio cue offset during replay
  - The difference in the length of the loading period w/o audio



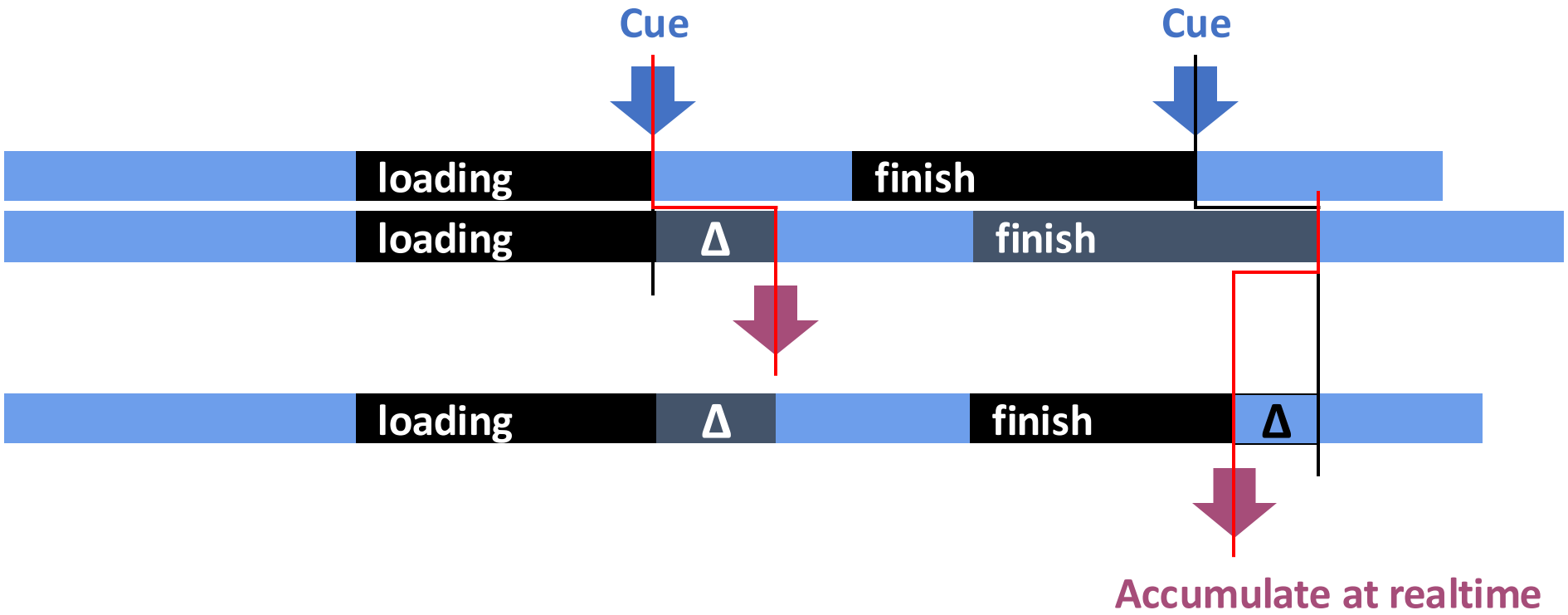
Audio Cues

- Apply the offset during replay
  - Shift the recording at realtime



Synchronization Cues

- Multiple cues
  - Accumulative
  - E.g., BeatSaber loading and finishing one song



Syncing Record and Replay: BeatSaber

- Full song 5 times replay score (1:50s)
  - **~0.8%** score error
    - Possibly due to the angle and timing of hitting

Combo	Score
94	27085
94	27411
94	27354
94	27877
94	27157

Alternative Cue: Visual / Text


```
>>> result = reader.readtext('0.jpg')
>>> print(result)
[([[709, 1149], [926, 1149], [926, 1269], [709, 1269]], 'COMBO', 0.8404282456039394), ([[3171, 1125], [3252, 1125], [3252, 1215], [3171, 1
215]], 'X', 0.42606678635204176), ([[3237, 1128], [3318, 1128], [3318, 1281], [3237, 1281]], '1', 0.14692433240624325), ([[774, 1272], [84
6, 1272], [846, 1377], [774, 1377]], '0', 0.6033891273898782)]
>>> result = reader.readtext('1.jpg')
>>> print(result)
[([[752, 1166], [970, 1166], [970, 1293], [752, 1293]], 'COMBD', 0.5119907731026628), ([[3208, 1108], [3380, 1108], [3380, 1286], [3208, 1
286]], 'x1', 0.5889345875864008), ([[813, 1293], [894, 1293], [894, 1398], [813, 1398]], '0', 0.42390021511279485)]
>>> result = reader.readtext('2.jpg')
>>> print(result)
[([[751, 1169], [964, 1169], [964, 1290], [751, 1290]], 'COMBD', 0.5508570132858225), ([[3202, 1111], [3371, 1111], [3371, 1286], [3202, 1
286]], 'x1', 0.7704156912772843), ([[803, 1295], [889, 1295], [889, 1402], [803, 1402]], '0', 0.8590763739038074)]
```

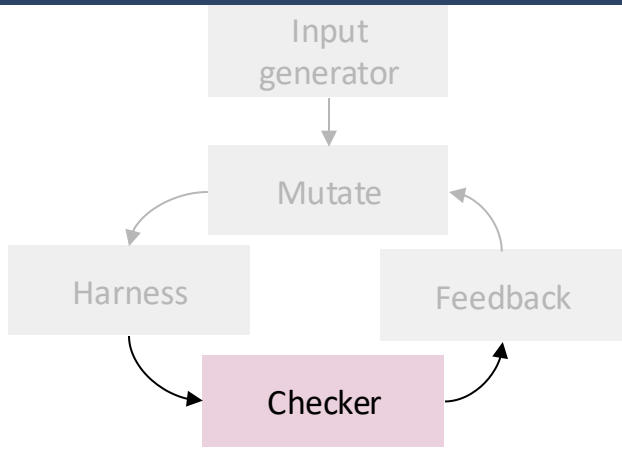
Other Synchronization Cues

- We can provide a group of general cues exist in VR for convenience
  - Audio
  - Fading (level transition)
  - Haptic
  - Text
  - Overlay (e.g., loading screens sometimes are presented as overlay)
  - Game specific, which we can provide API for user to define

Abnormality Detector

Performance degradation

Runtime errors

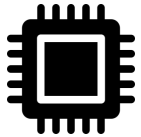


Motion sickness

Chaperone trespass

Excessive Performance Degradation

The most noticeable impact on VR QoE



System Performance Metrics

- CPU and memory utilization
- CPU and memory power consumption
- GPU graphics and memory utilization
- GPU graphics and memory clock speed
- GPU power consumption
- GPU encoder and decoder usage

VR Runtime Metrics

- Frame rate
- Dropped frames
- Reprojected frames
- Reprojection ratio
- Numbers of timeout

$$\Delta\left(\text{Expected behavior from previous input}, \text{Performance status from current input}\right) > \varepsilon$$

Unexpected Runtime Events

- VR events recorded with inputs
- Compare behaviors between current and previous replays
  - Frequency and characteristics of VR runtime events
  - Detect abnomalies or excessive occurrences

VR runtime events

- VREvent_Input_HapticVibration
- VREvent_Compositor_ChaperoneBoundsShown

…

VR runtime errors

- EVRCompositorError
- VREvent_TrackedDeviceDeactivated
- VREvent_ProcessDisconnected

…

Motion Sickness

Display lag (motion-to-photon latency) has significant impact

Δ > 20 ms

Mirage: Δ (input, frame reprojection)
Alert cybersickness risks if Δ > 20 ms

Chaperone Trespass

Chaperone: VR boundary alerting players to real-world obstacles

→ Lack of trespass warnings poses significant **physical risks**

Mirage monitors when
a warning should be triggered
but it has not.

If (Pose(player) > Chaperone) &
!(Warning || Passthrough):

Alert

Feedback

Gathered from the same sources as the abnormality detector

Guide the mutation engine to prioritize inputs

System Performance Metrics

- CPU and memory utilization
- CPU and memory power consumption
- GPU graphics and memory utilization
- GPU graphics and memory clock speed
- GPU power consumption
- GPU encoder and decoder usage

VR Runtime Metrics

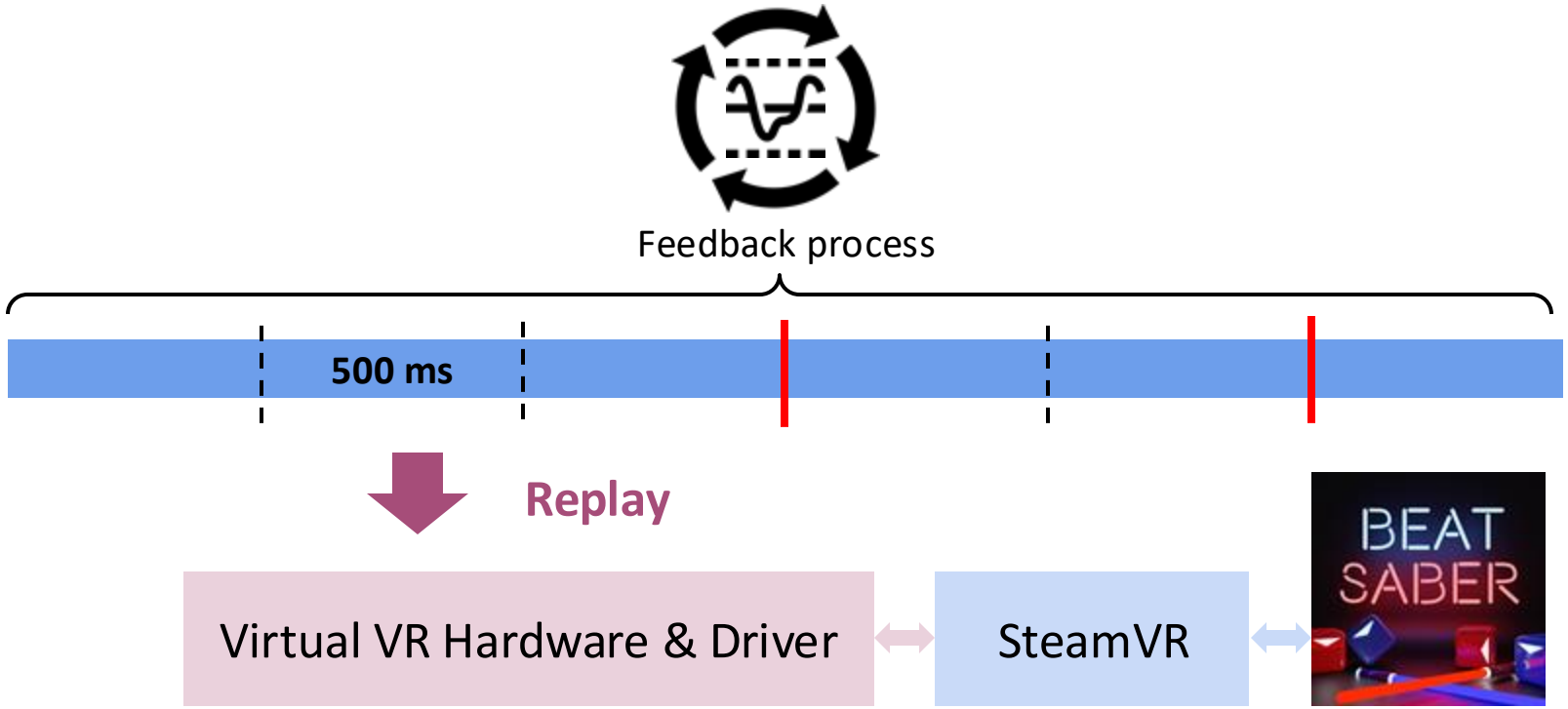
- Frame rate
- Dropped frames
- Reprojected frames
- Reprojection ratio
- Numbers of timeout
- Number of warning / error events
- Guardian boundary / Proximity warning

Performance degradation → lower QoE score

More/new errors and warnings → lower QoE score

Collecting Feedback

- Record “interesting” moment (i.e., timestamps)
  - E.g., peak frame drop, peak GPU, passing threshold, etc.
- Returns a vector of interesting moments to mutate

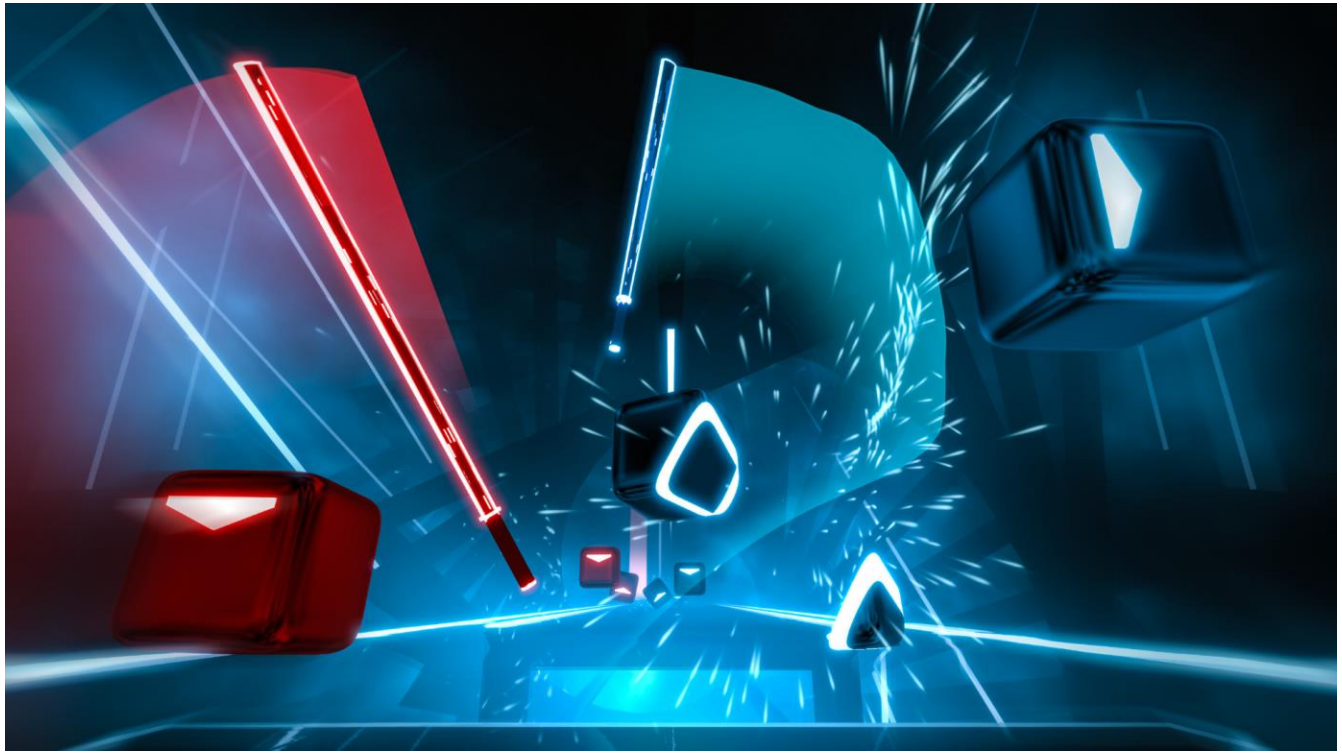


Case Study: BeatSaber

- Preparation recording
  - A lobby
  - Select a particular song
  - Optionally configures
- Returns to the same lobby after a session ends

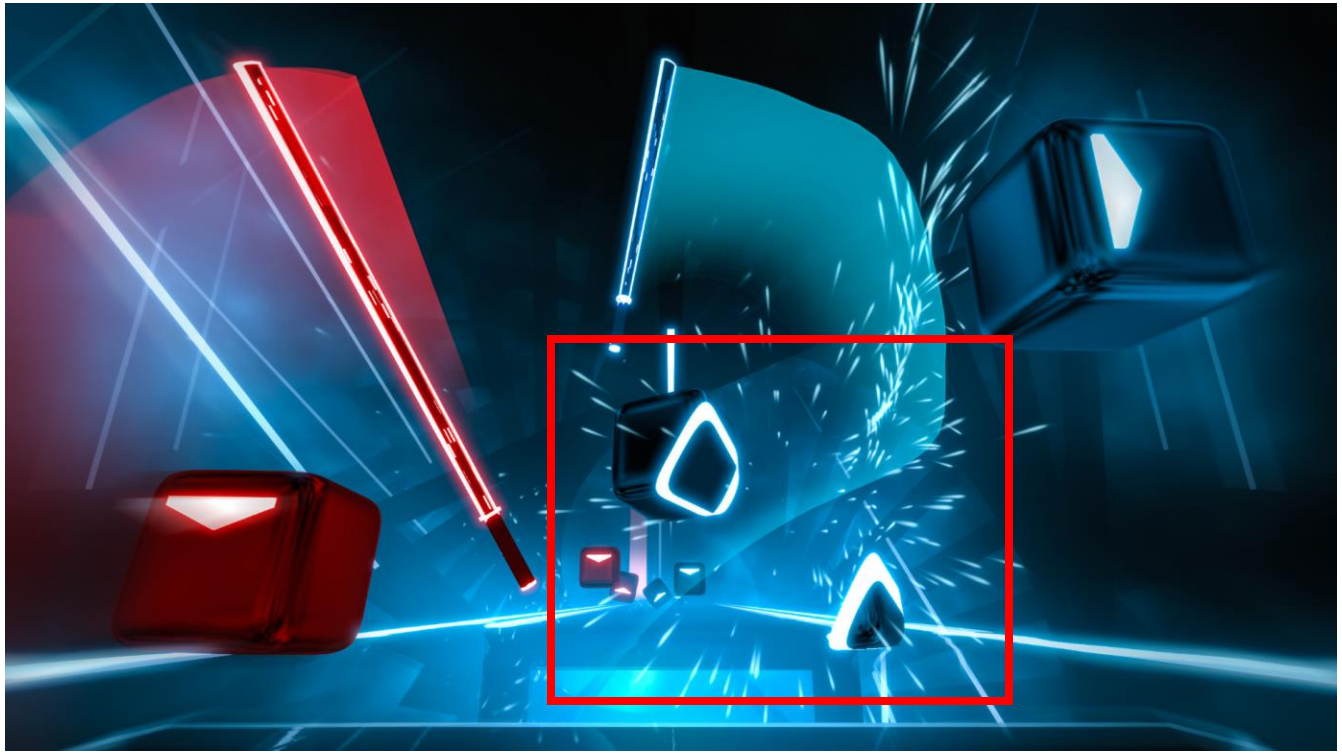
Case Study: BeatSaber

- Main testing scene
  - Faces an oncoming stream of colored blocks that align with the rhythm
  - Slash blocks in specific directions indicated by arrows
  - Obstacles such as walls that the player must dodge



Case Study: BeatSaber

- Seed inputs
  - A human player playing a specific song from start to finish
  - For testing purposes, intentionally hit obstacles to trigger specific application logic that handles such interactions.



Case Study: Half-Life: Alyx

- Preparation recording
  - Main menu
  - Start game
  - Load game
  - Settings
- Mirage requires to progress to the desired testing scene and save the progress
  - Simply load the saved progress to prepare
  - Return to menu to end cycle

Case Study: Half-Life: Alyx

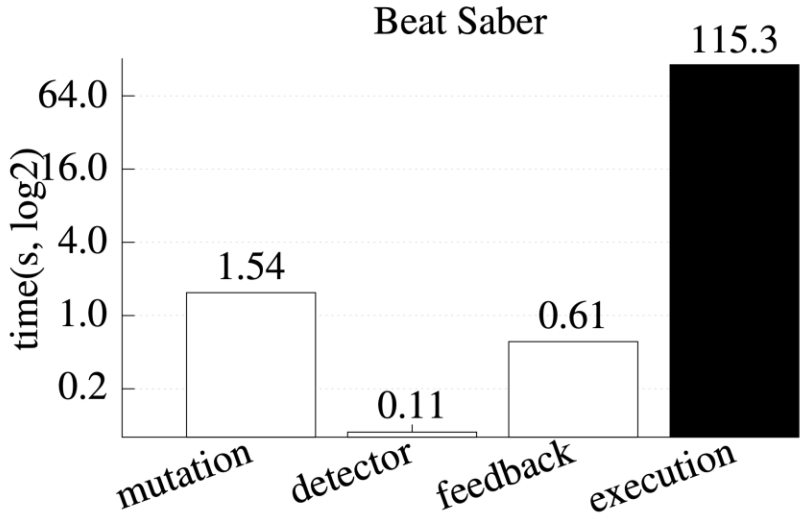
- Testing scene: Vance's apartment
- A variety of interactive objects
  - Tools and electronic components
  - Holographic displays
  - Maps and scientific equipment
- Deeply interactive VR tasks
  - Assembling devices
  - Examining plans
  - Manipulating 3D holograms.

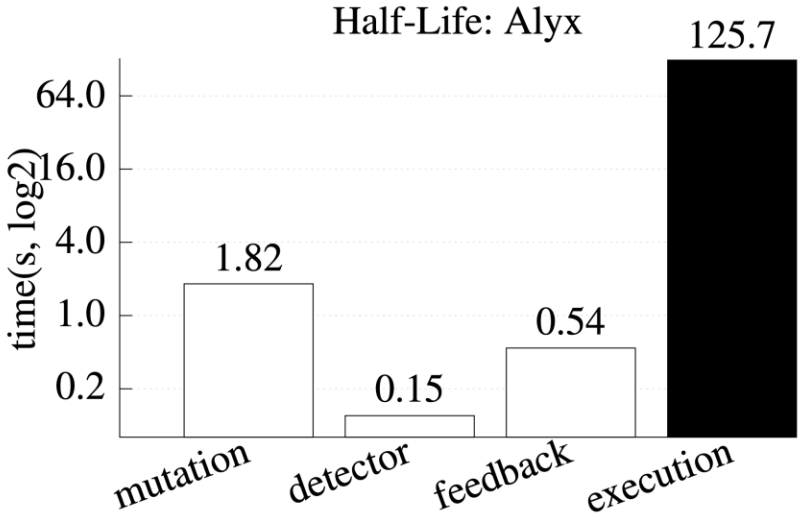
Case Study: Half-Life: Alyx

- Seed inputs
  - Moving in the environment
  - Interacting with objects
  - Using tools
  - Engaging in combat
- Capture the intricacies of player movement and decision-making
  - Angle of picking up an object
  - Method of solving a puzzle
  - Strategy used in combat

Performance Evaluation

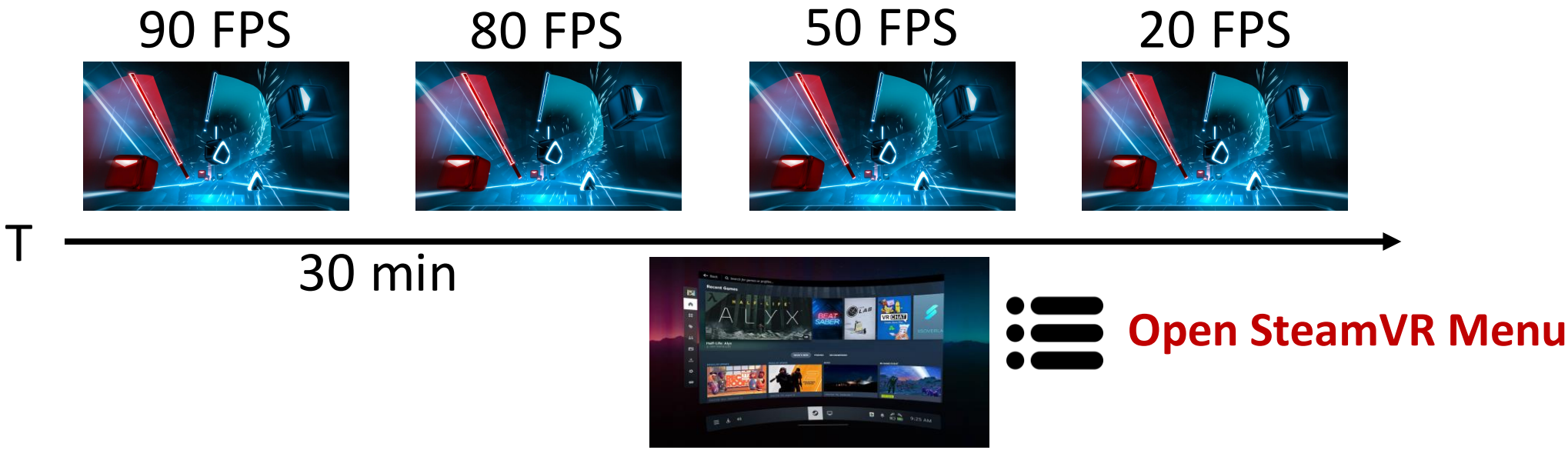
- Duration of a test is decided by the application and the length of inputs
  - Beatsaber: 115 seconds / test of the picked song (1m50s)
  - Half-Life: Alyx: 125 seconds / test (i.e., 5s preparation and 120s play)
- Mirage-specific components accounted for 2% of the total testing time
  - 98% of the testing time was spent on the VR application
- Crucial when testing for performance abnormalities





Case #1: Performance Impact Caused by Specific Interactions

Requirement: specific interactions

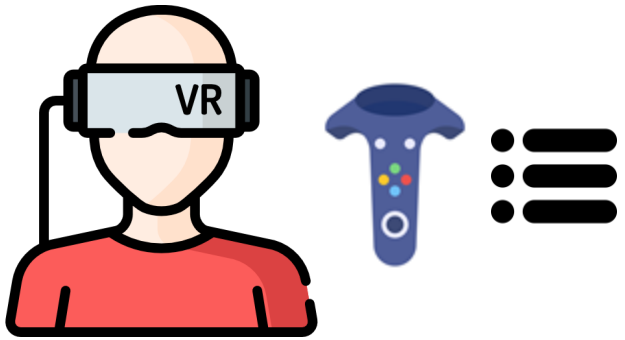


- Consequences
  - Reduced playability
  - Player discomfort (e.g., cybersickness)
- Potential root cause
  - Memory leaks

Triggering in Real World

- Challenging in conventional testing environments
- Diversity in real user behavior significantly increases the likelihood

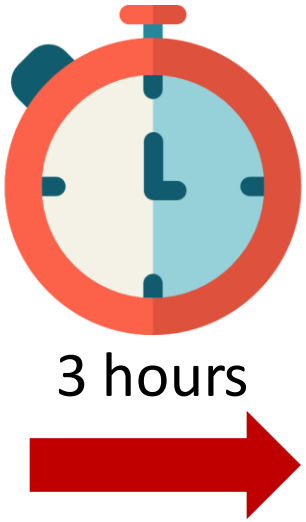
E.g., Players with aggressive movements

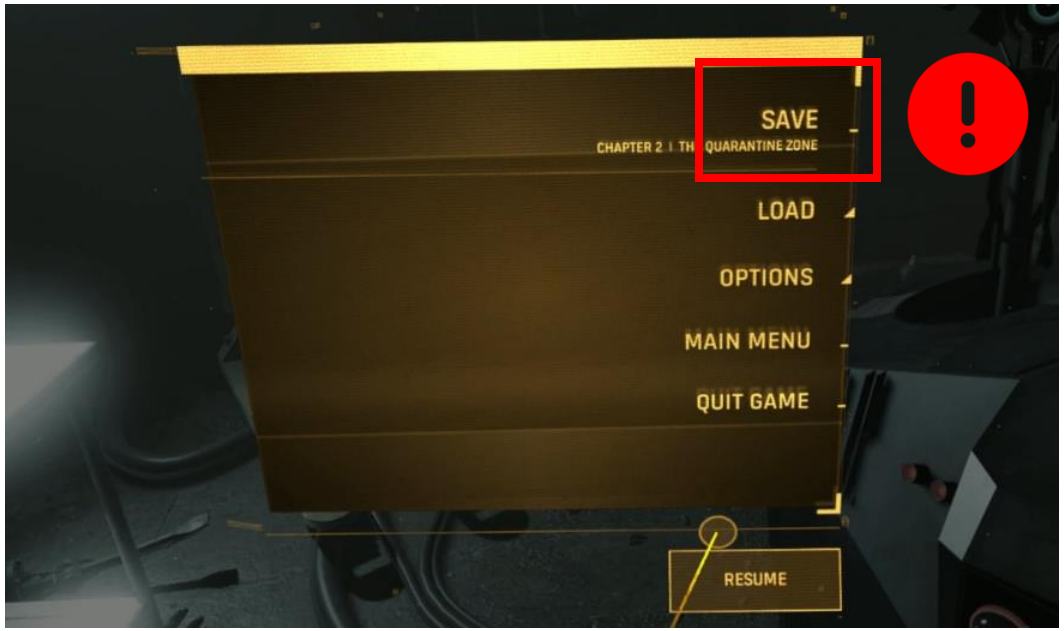


E.g., Players who frequently access SteamVR menu as strategy or habit

Case #2: Crash Caused by Long Interaction Sessions

Requirement: long interaction sessions

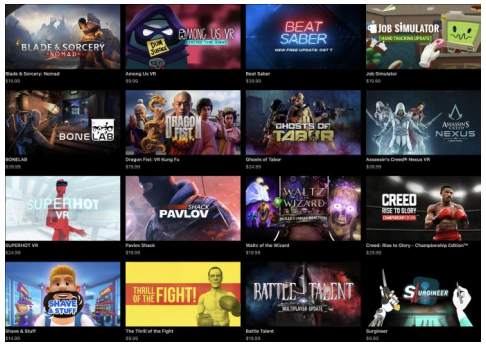




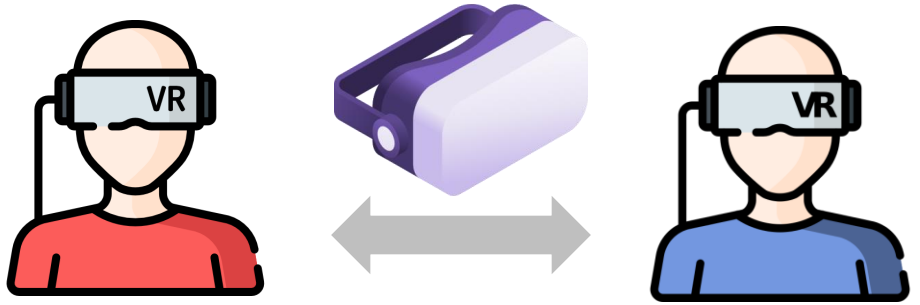
- Consequences
  - Frustrate players
  - Risk disengagement
  - Harm reliability reputation
- Potential root cause
  - Resource exhaustion
  - Overload the saving system

Triggering in Real World

- Manual testing is limited in duration due to practical constraints
  - Tester fatigue and resource availability
- Automated tests often focus on shorter and more discrete scenarios
  - Easily repeated for efficiency
- Issues can emerge only after several hours of continuous operation



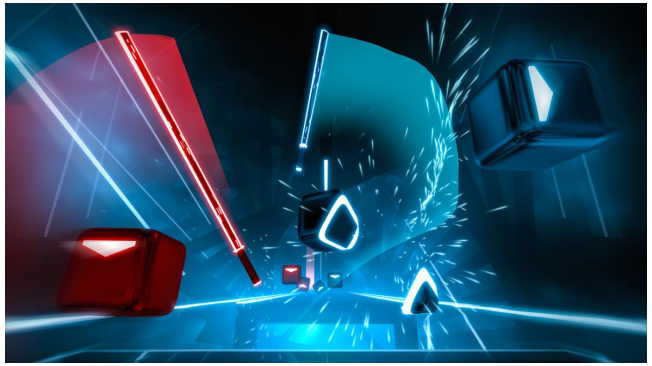
E.g., COTS VR applications

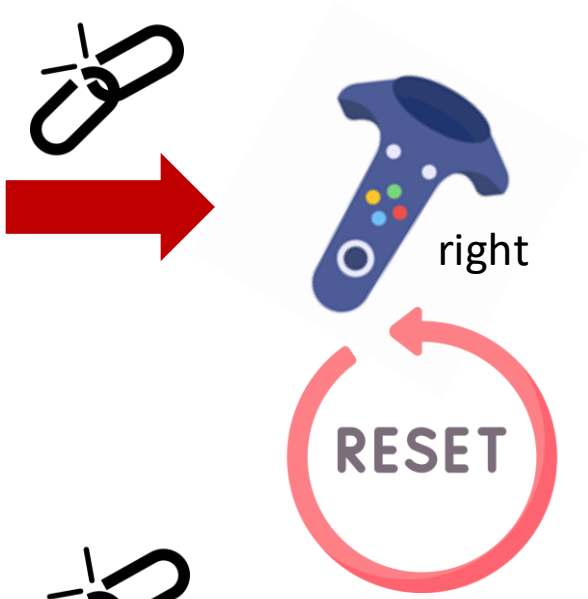


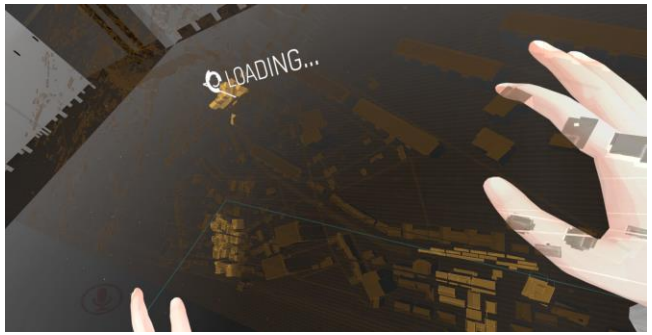
E.g., Shared device

Case #3: QoE Impact Caused by Device Management

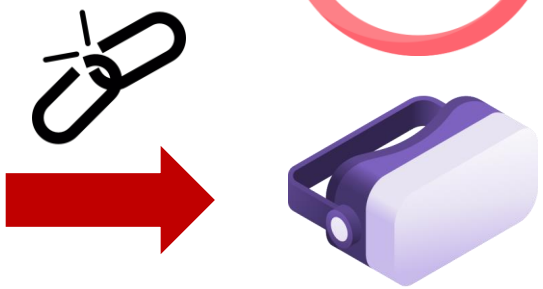
Requirement: device management







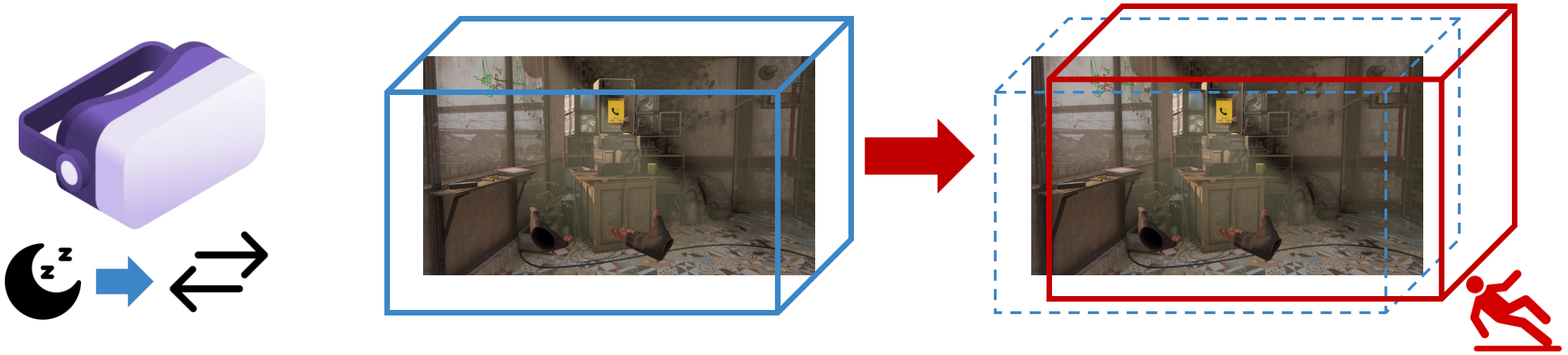
Half-Life: Alyx Loading Screen



- Consequences
  - Disrupt immersion
  - Practical annoyances
  - Reducing perceived reliability of the VR platform
- Potential root cause
  - Inadequate management of device status changes
  - Flaws in reconnection protocols within the applications

Case #4: Physical Risk Caused by Device Management

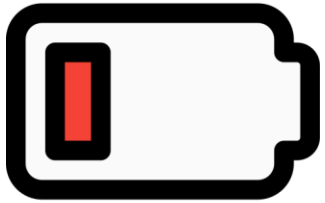
Requirement: device management



- Consequences
  - Confuse real spatial limits
  - False impression of safe zone
  - Risk of colliding
- Potential root cause
  - Flaws in the chaperone management of the application

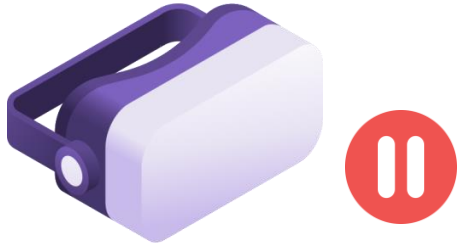
Triggering in Real World

- Conventional VR testing does not include VR runtime events
  - Code-level testing lacks VR system context
- Real-life VR devices frequently encounter device sleep or disconnection



E.g., Low battery levels

E.g., Noisy wireless signals



E.g., Temporarily remove the HMD

Discussion and Future Work: Seed Input Pool

- Strawman approach – record from physical device
  - The only manual part
- Emerging VR interaction datasets
  - 50,000+ Virtual Reality Users of BeatSaber
  - 4.7 Million Motion Capture Recordings from 105,852 Users of BeatSaber
  - 110+ hours of motion, eye-tracking and physiological data from 71 players of Half-Life: Alyx
- Use as input seeds
  - Adapt to Mirage input

Discussion and Future Work: ML-Assisted Testing

- ML-assisted input feedback and mutation
  - Feedback values are system and VR runtime specific, not application specific
  - Mutation does not incorporate application semantic
  - GPT4 with Vision and GPT4-o
- ML-based oracle
  - ML-based cybersickness oracle
    - Machine learning architectures to predict motion
    - Cybersickness prediction from integrated hmd's sensors
    - Truvr: Trustworthy cybersickness detection using explainable machine learning
  - ML-based QoE oracle
    - Questset: A VR Dataset for Network and Quality of Experience Studies
- Performance limitation of current ML solutions

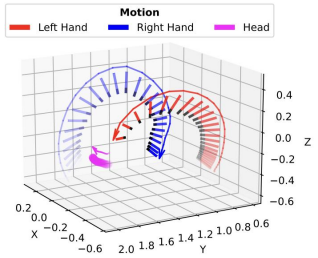
Mirage Summary

- Virtual driver for automation without physical devices
- Input synchronization
- Mutation engine, abnormality detector, and QoE feedback
- Take aways
  - Confidentiality: The interactive characteristics uniquely identify users
  - Integrity violation can be imperceptible
  - Users are less tolerant to impacts on QoE (immersiveness) vs. traditional bugs

How to adapt TEEs for VR to ensure confidentiality and integrity while maintaining QoE?

Motivation – Confidentiality

Body motions

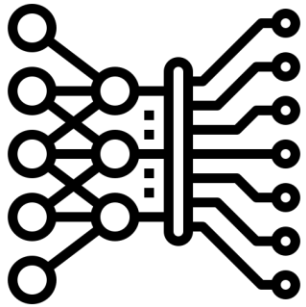


Eye gaze

Hand joints

Facial expressions
…

Personal information



Data processing
and machine learning

Identity

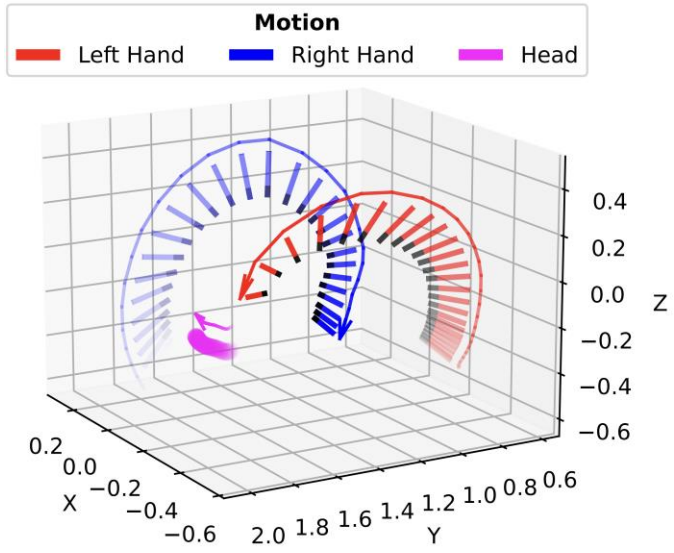
Gender
Age
Medical
…

Proximity sensors
Passthrough
Point cloud

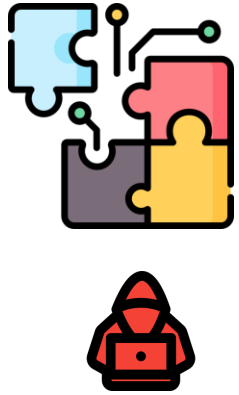
Spatial context

View and topology
of surrounding

Motivation – Integrity



User inputs



Clickjack

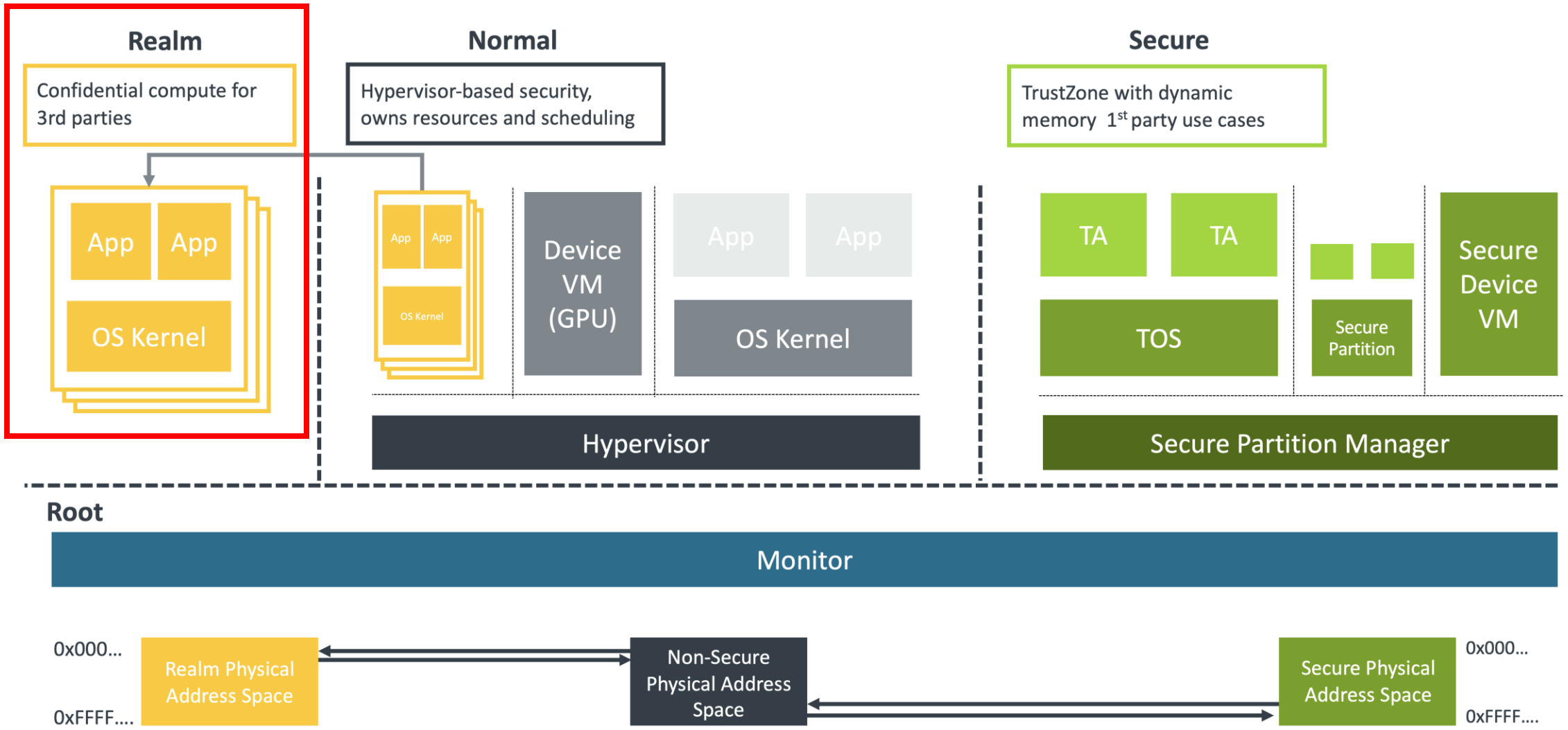
Ad impression profit

Unintended actions

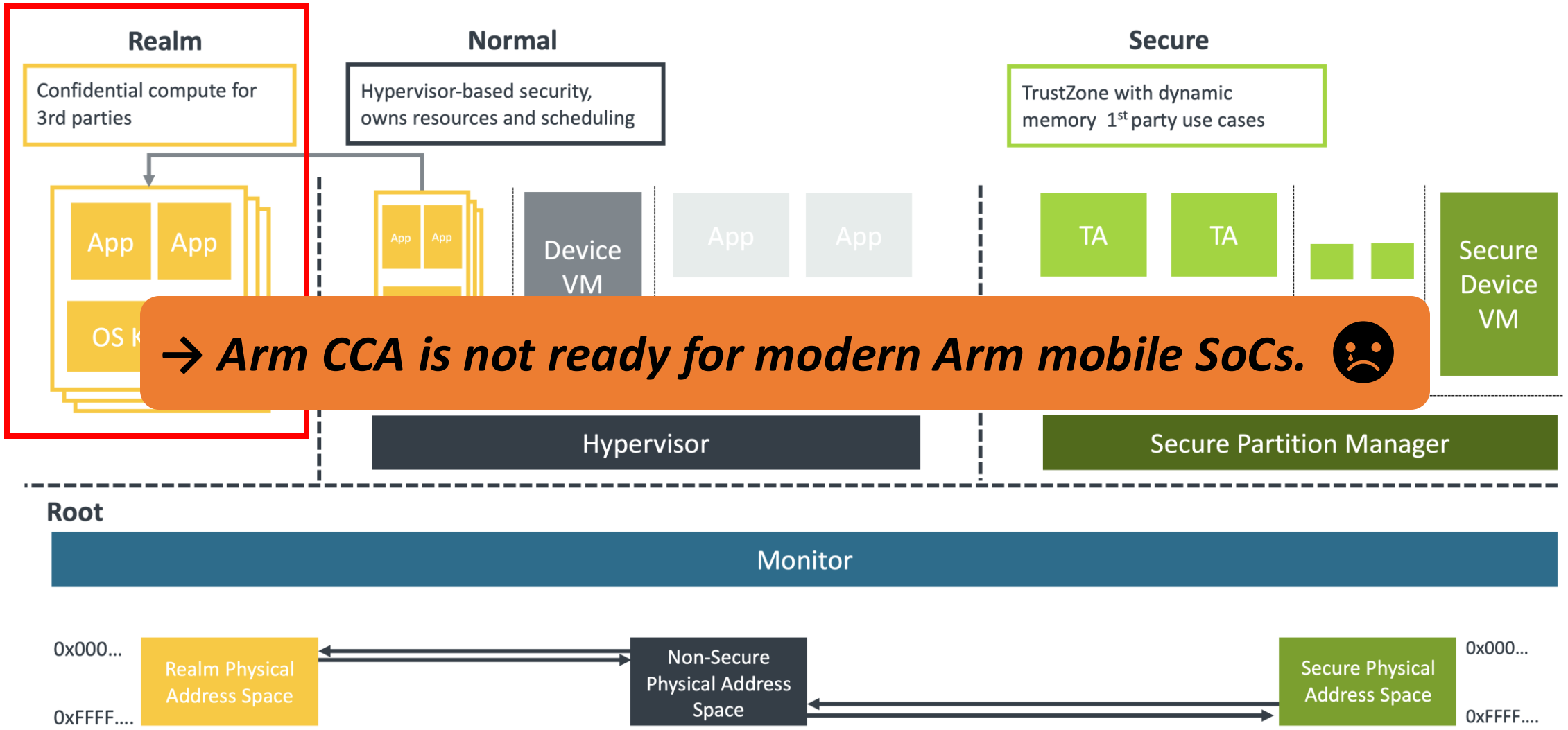
Injury

...

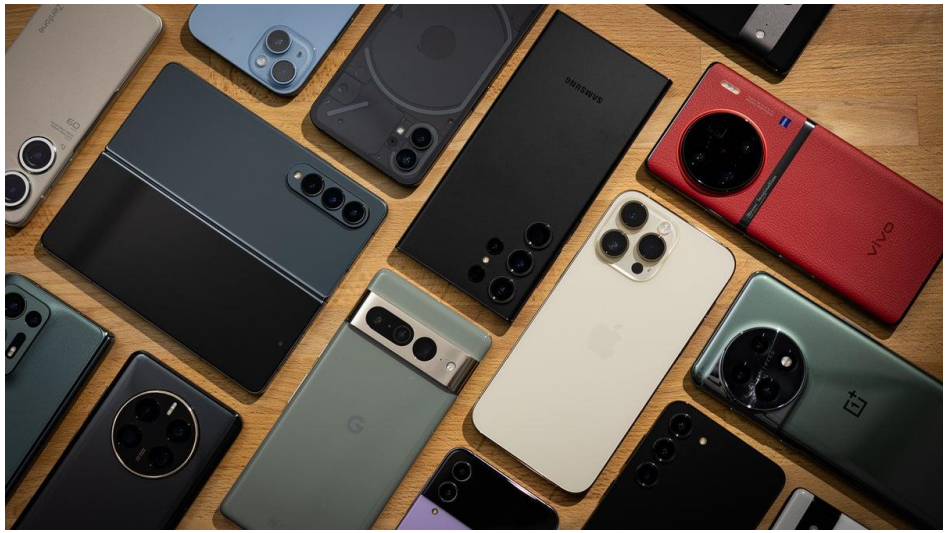
Arm Confidential Compute Architecture (CCA)

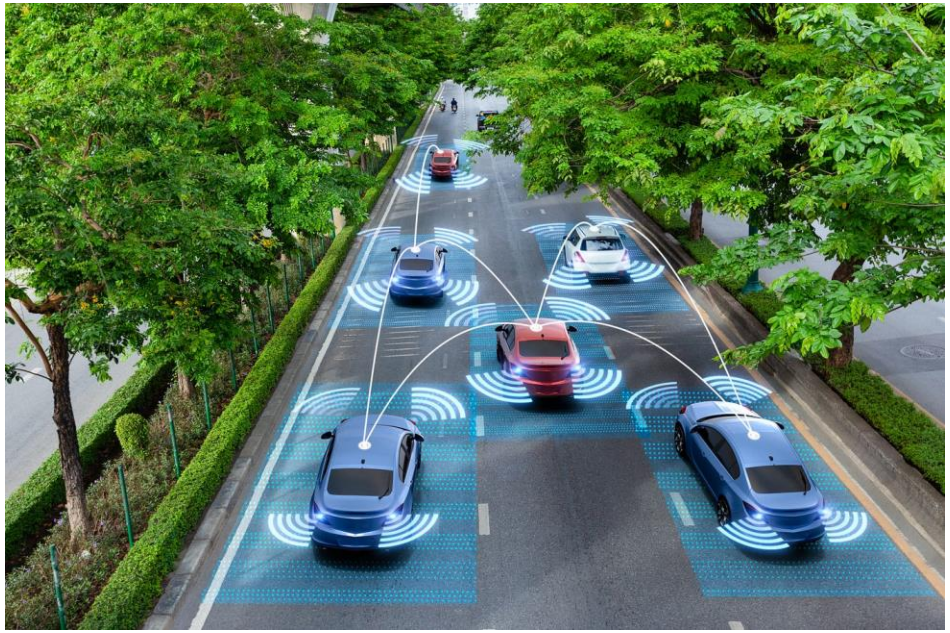


Arm Confidential Compute Architecture (CCA)



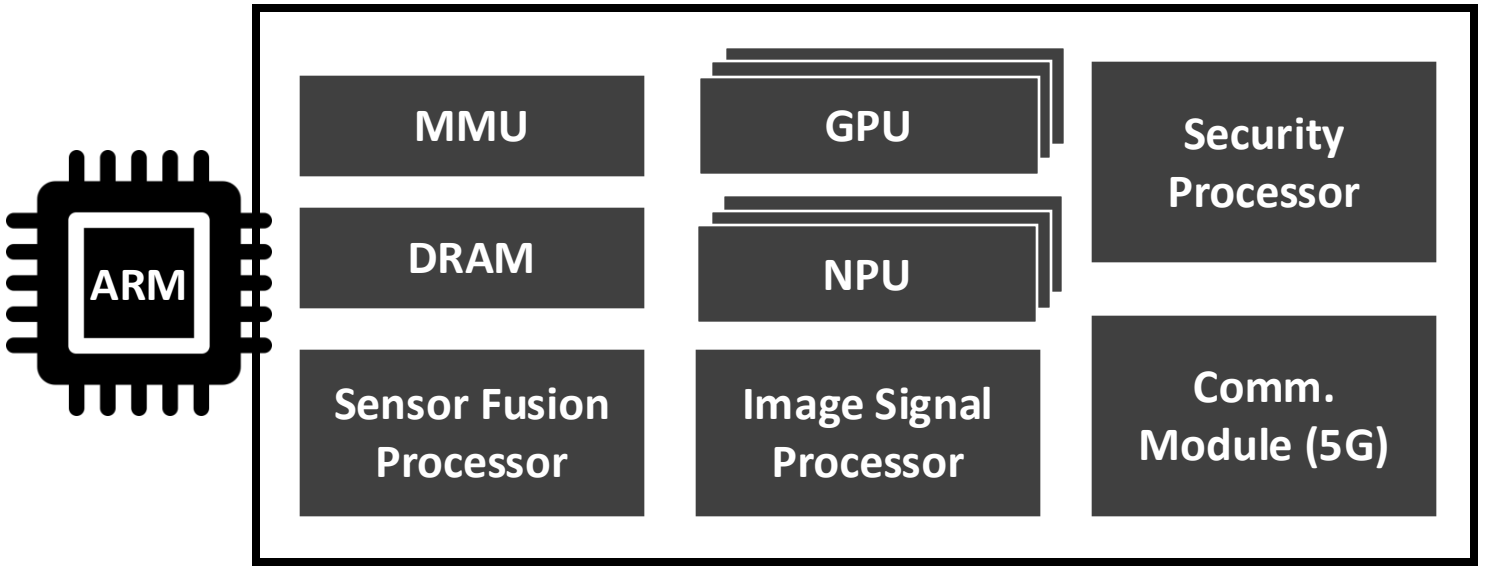
Arm – 99% Market Share in Mobile

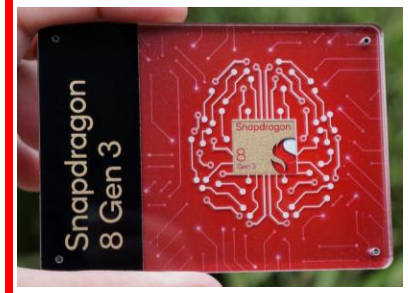


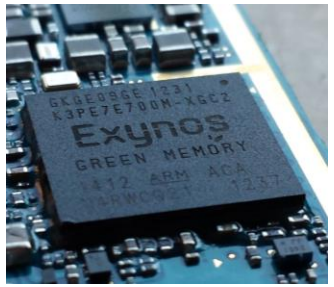


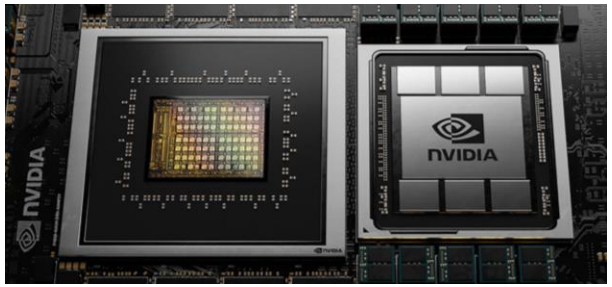
Arm – Architectural Trend in Mobile SoCs

An increasing integration of devices

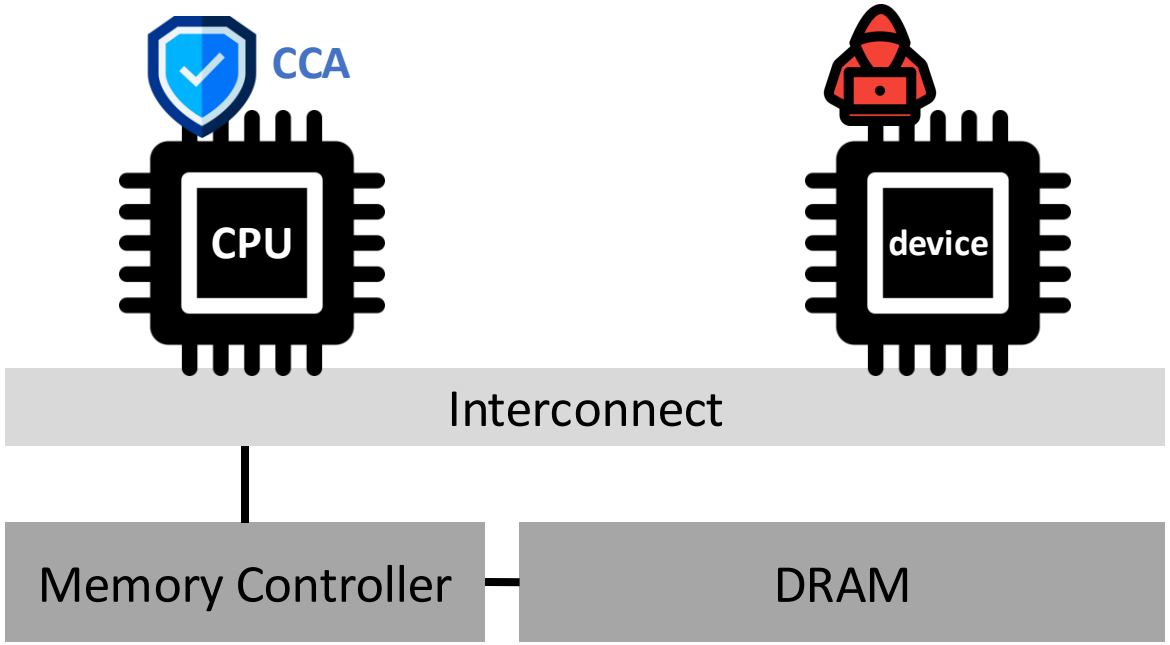




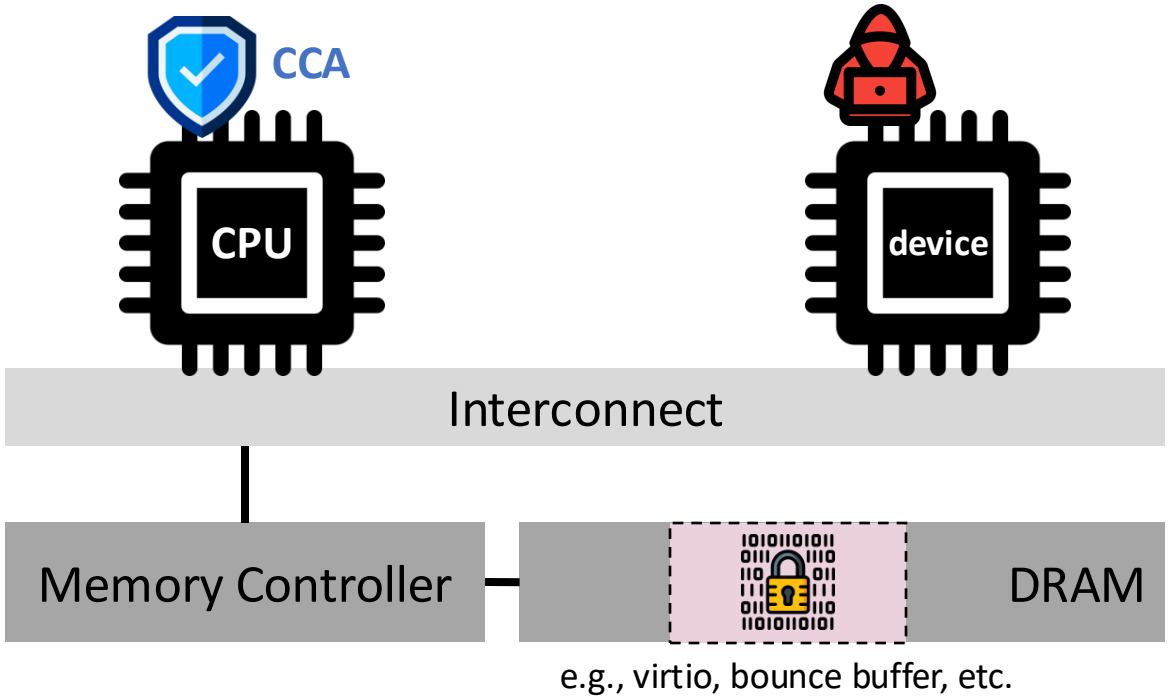




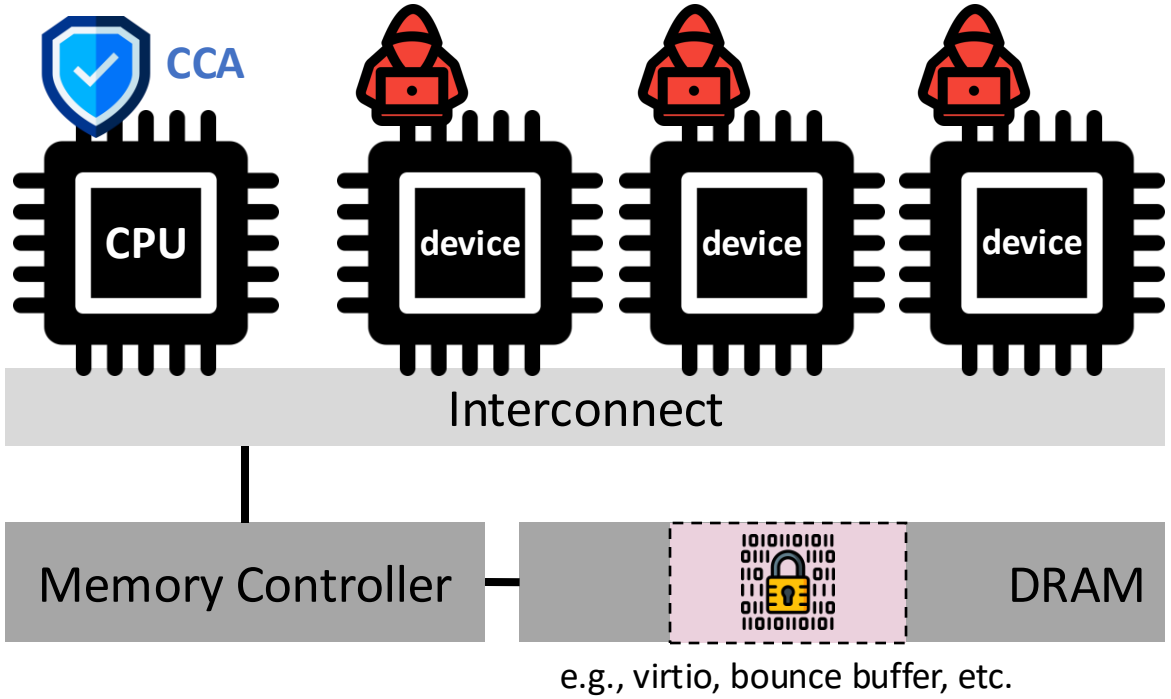
Device Access in CCA



Device Access in CCA

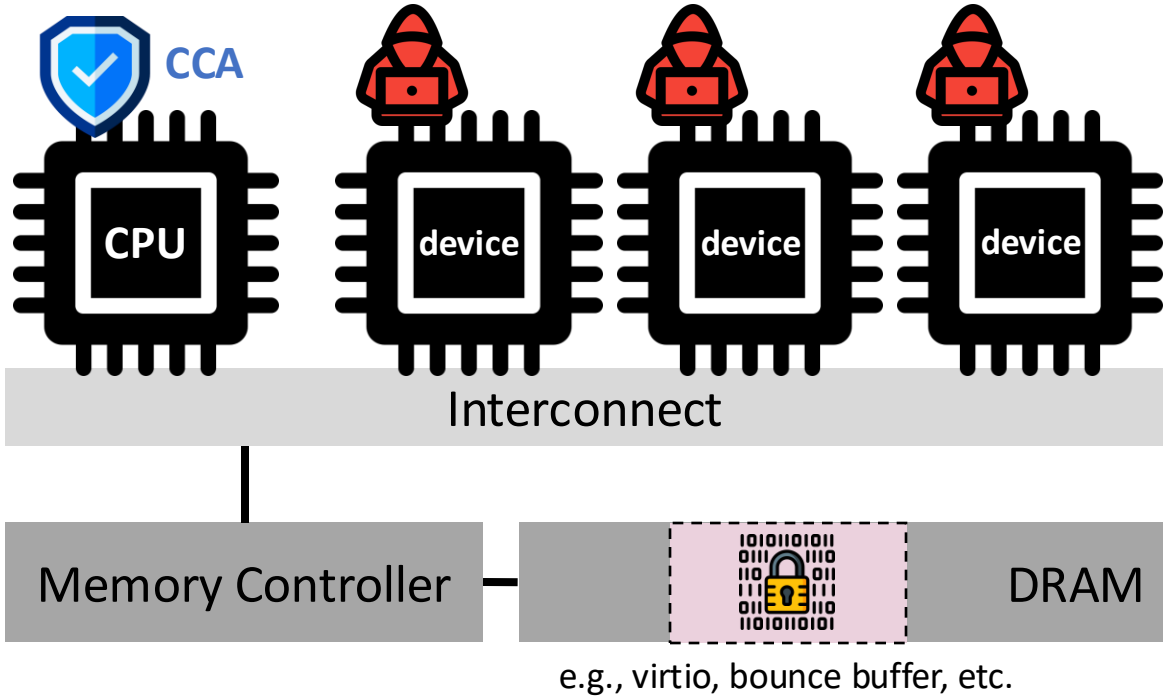


Device Access under the Architectural Trend



→ ***Arm CCA struggles to keep up with the architectural trend.***

Device Access under the Architectural Trend



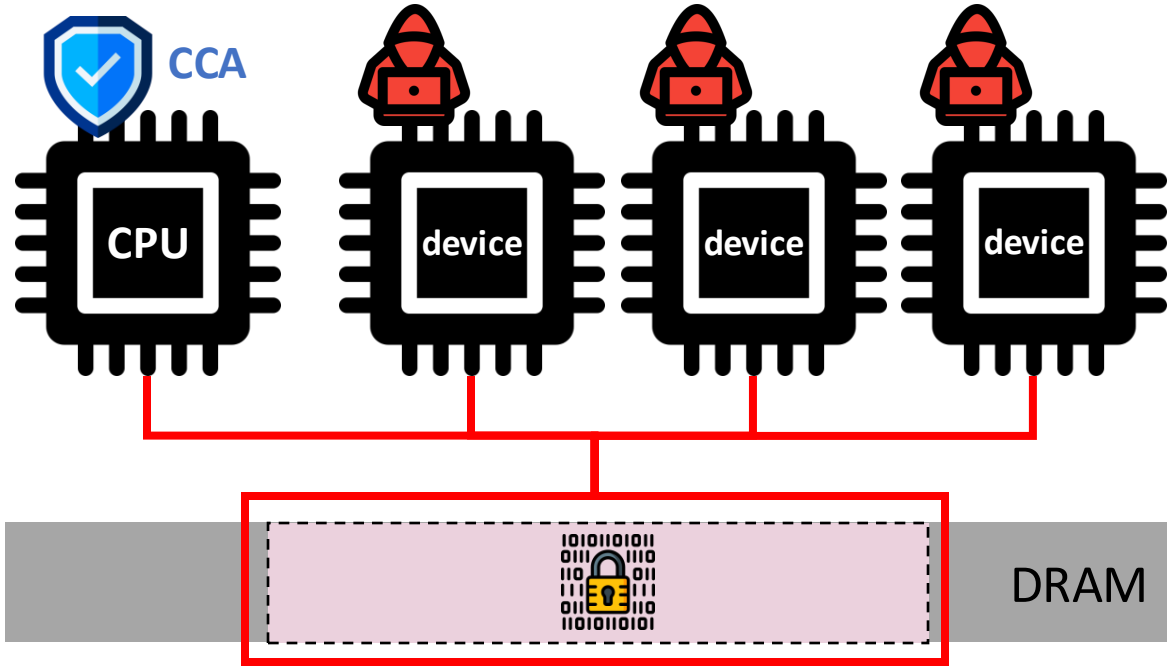
+ Robust security
- I/O performance and scalability
- Dynamic device management
- Power efficiency

→ ***Crucial for mobile platforms***

→ ***Crucial for CCA wide adoption***

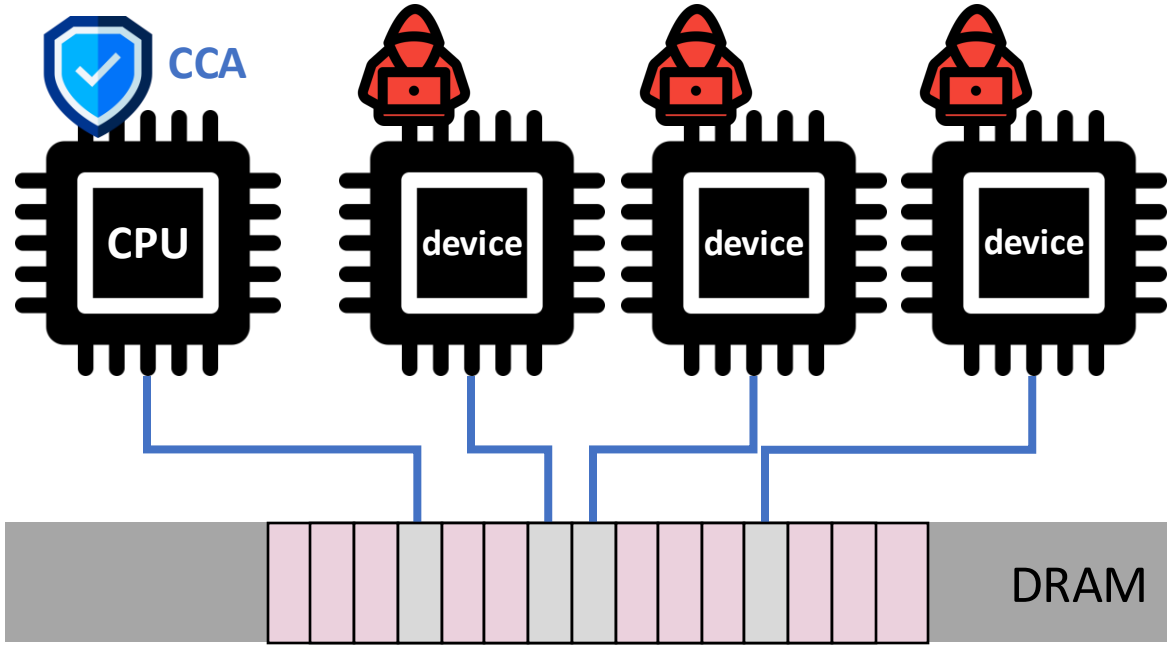
→ ***Arm CCA struggles to keep up with the architectural trend.***

Device I/O Performance and Scalability



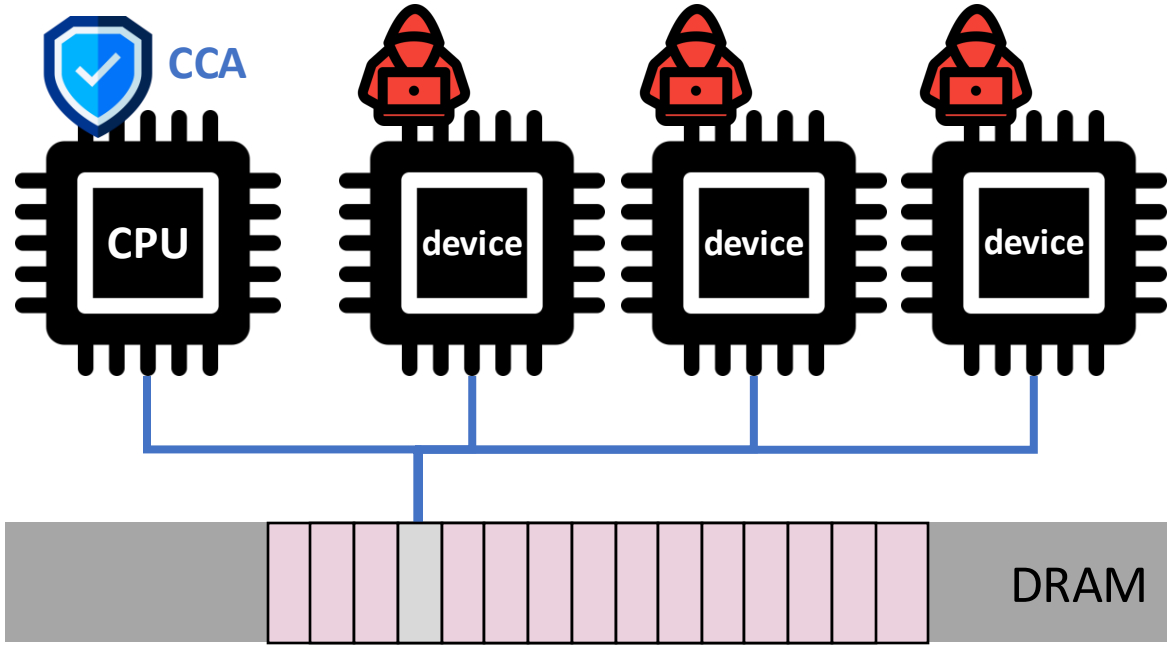
Encryption and decryption for each memory access

Device I/O Performance and Scalability



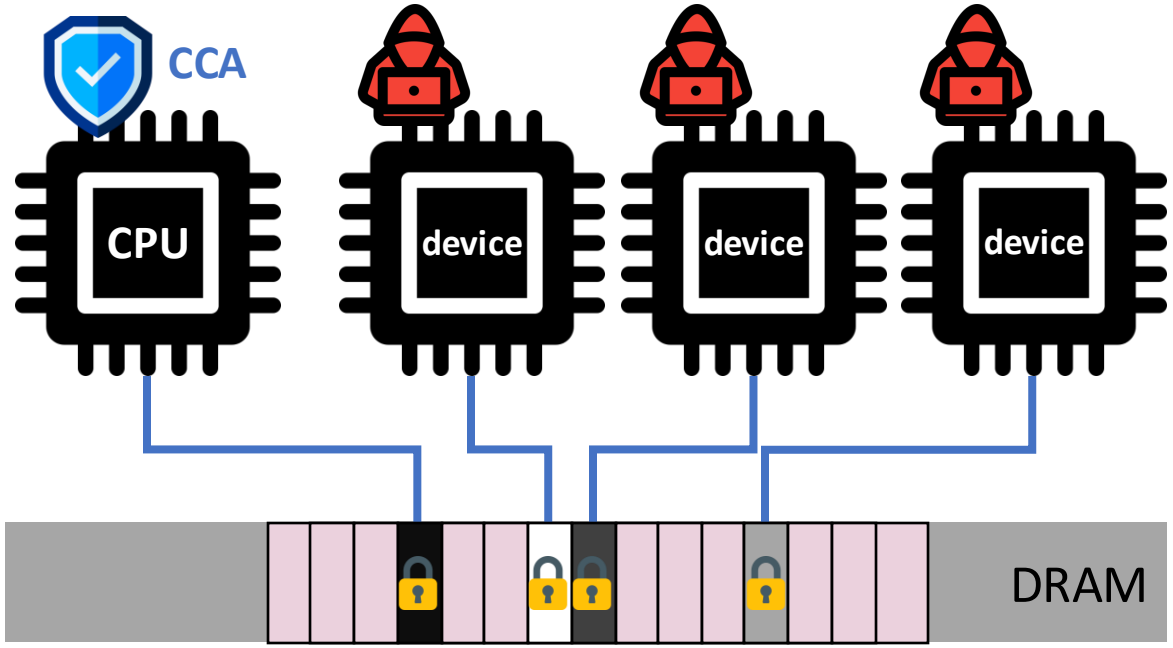
Memory Deduplication

Device I/O Performance and Scalability



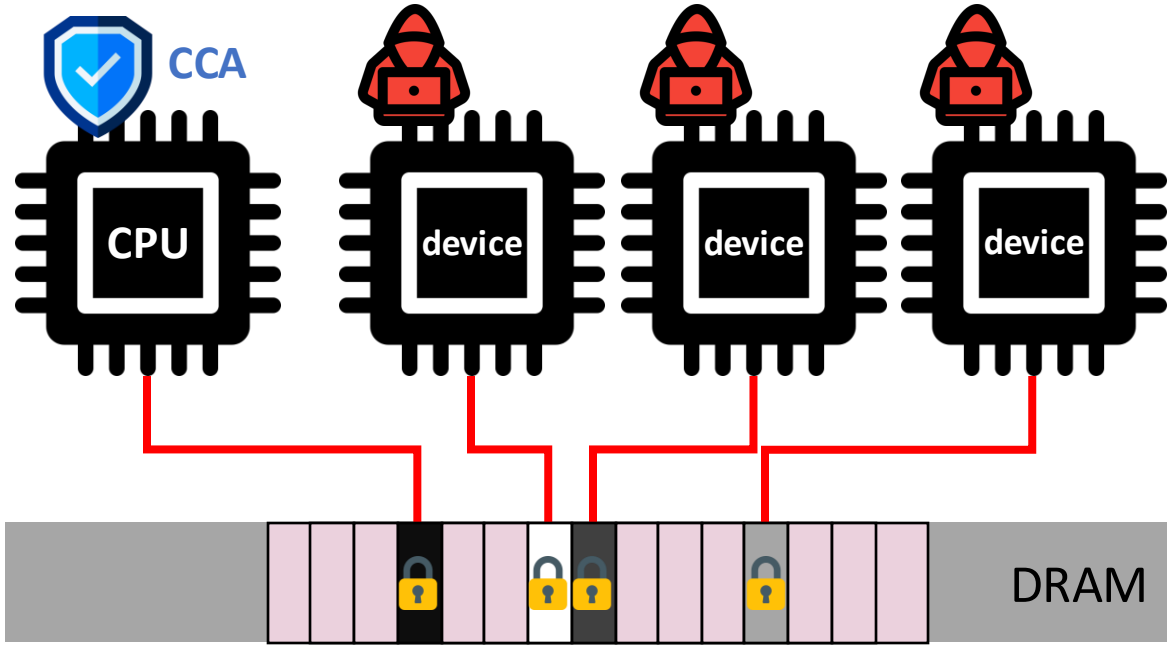
Memory Deduplication

Device I/O Performance and Scalability



Memory Encryption

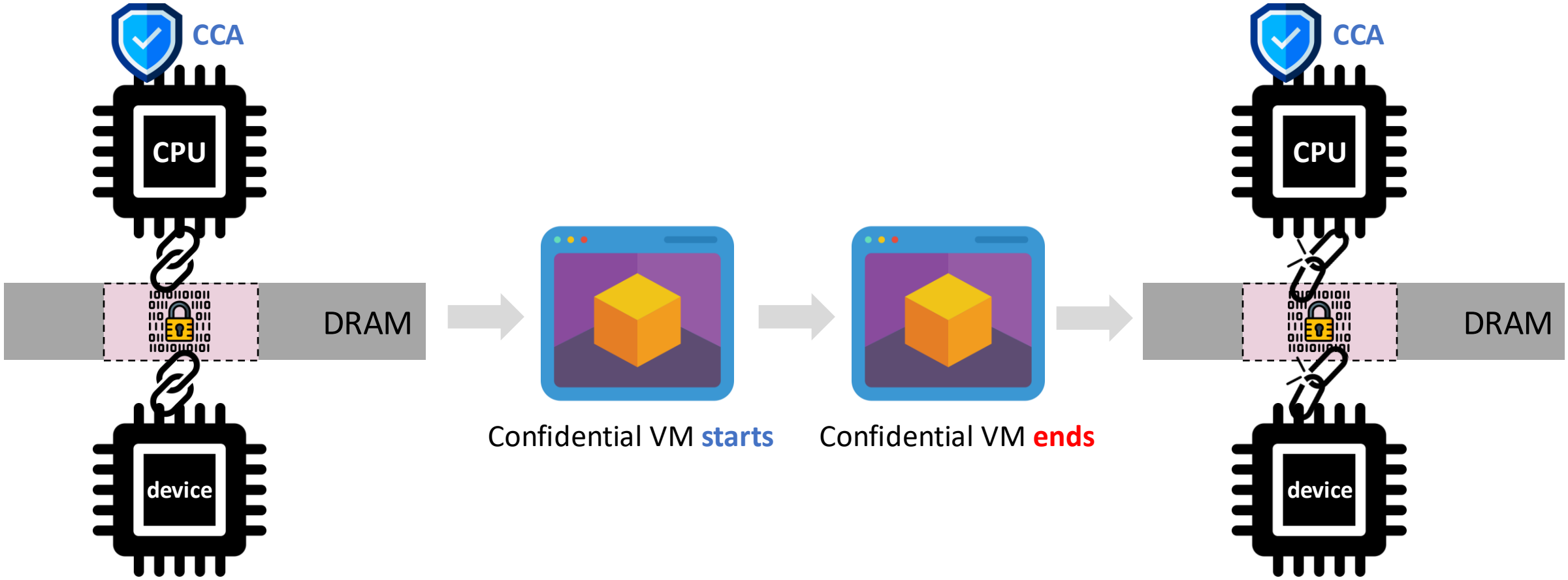
Device I/O Performance and Scalability



Memory Deduplication

Dynamic Device Management

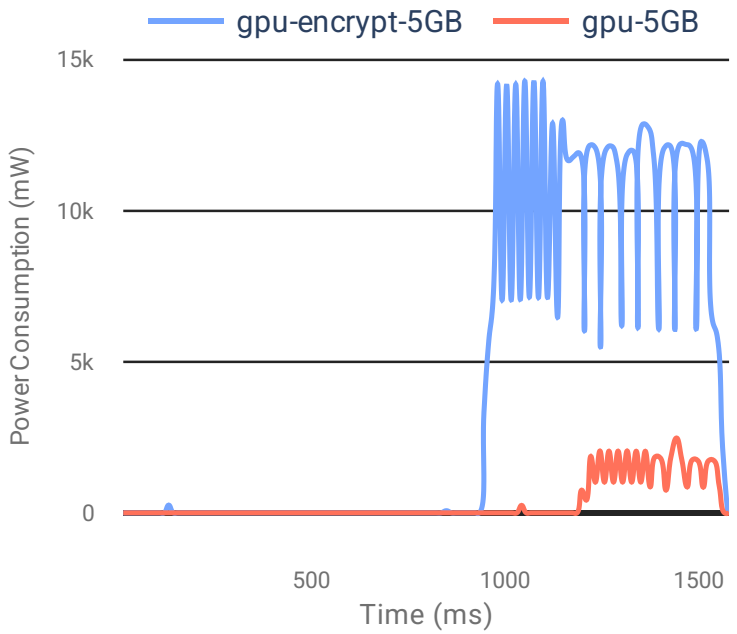
Device is bond to the entire lifespan of the confidential VM



Power Efficiency

Overhead incurred by memory encryption on mobile SoC

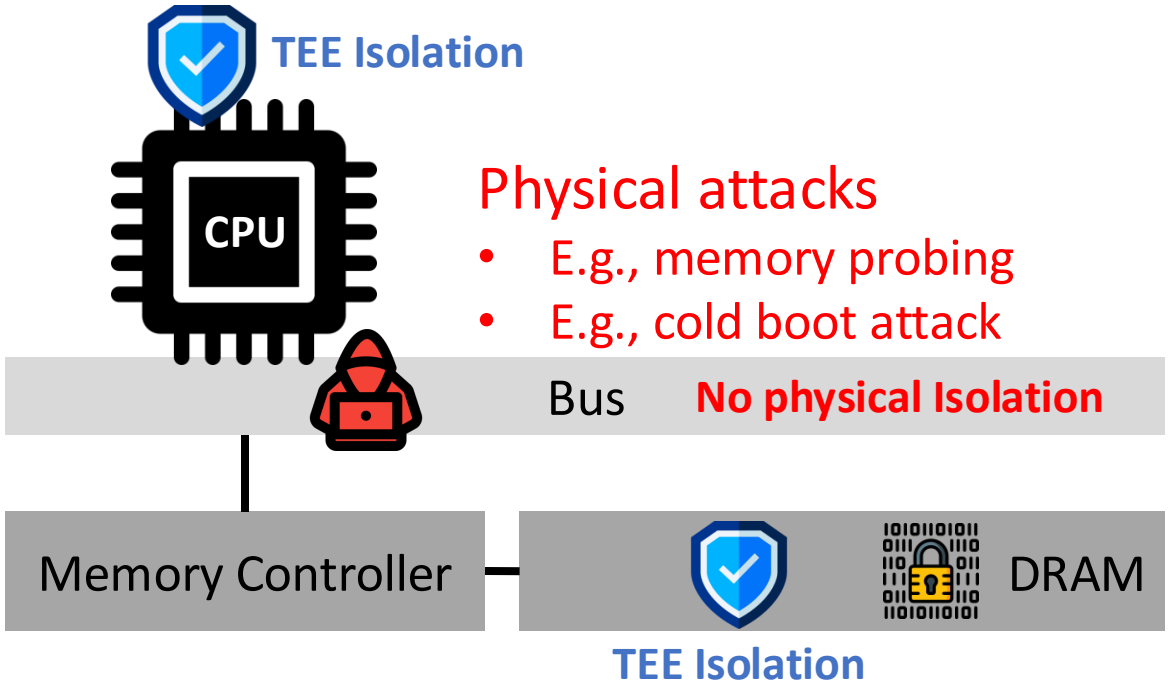
- Increased power usage
- Decreased throughput



Copying 5GB of memory with and without encryption

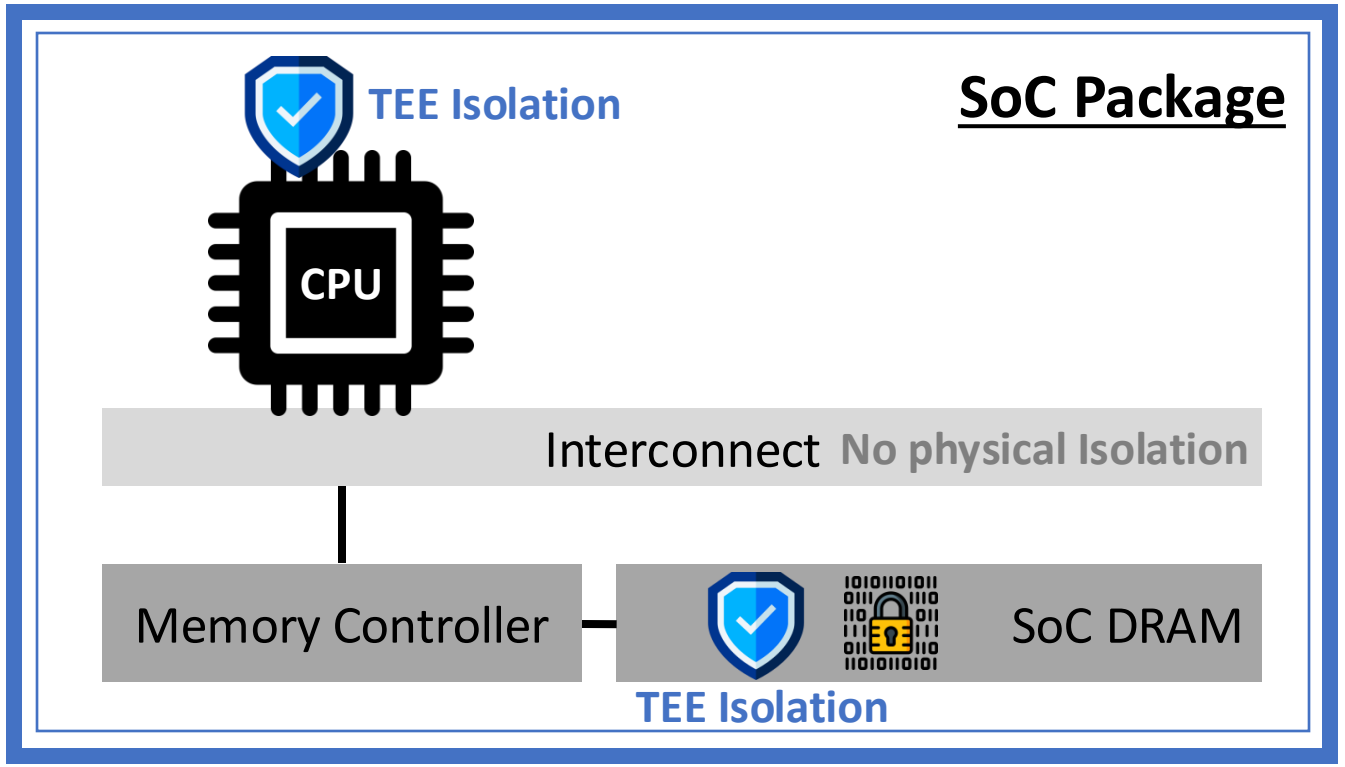
128-bit AES-GCM on Apple M1 SoC with 16GB unified memory

Memory Encryption in TEEs

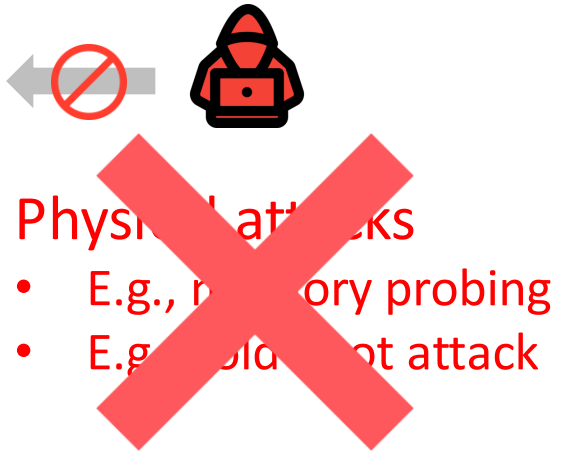


Mobile Arm SoC

Integrated Memory in SoC



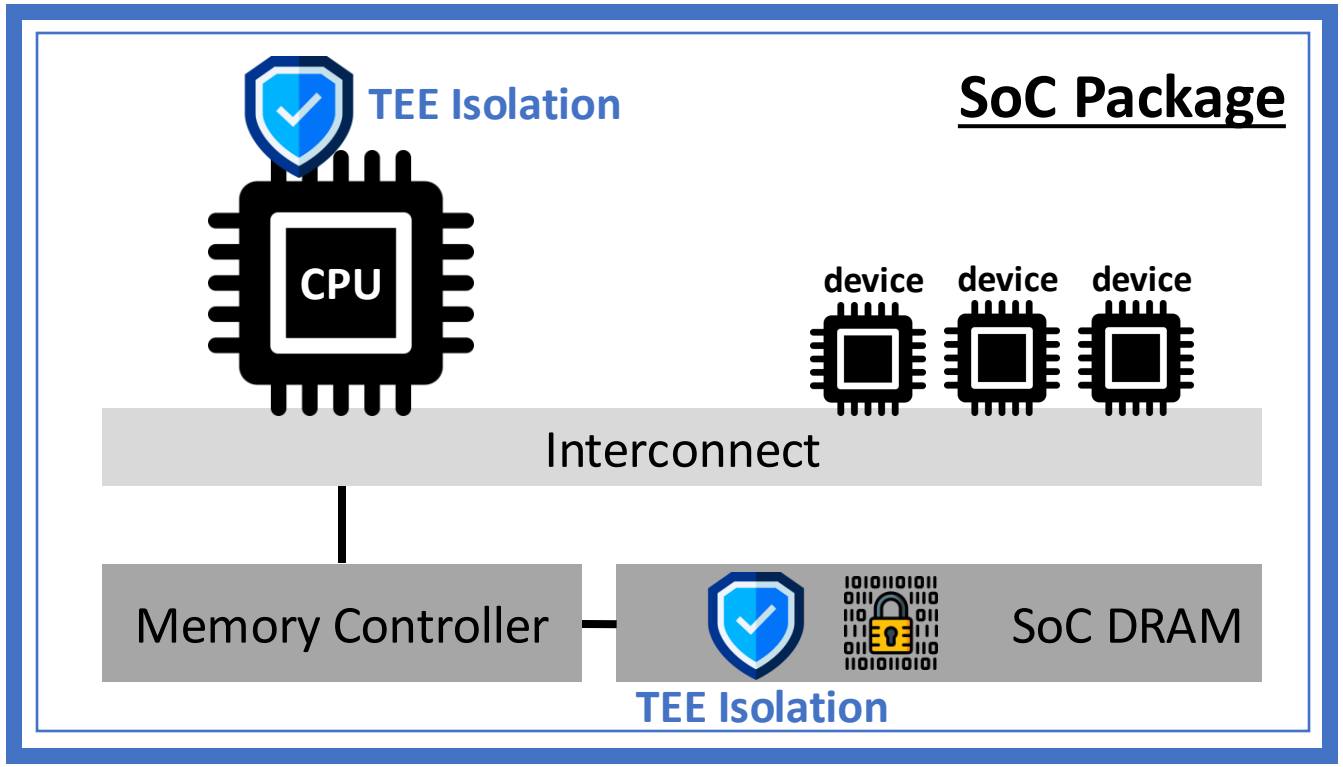
- Compact and integrated
- Advanced packaging technology
- Security features
  - E.g., tamper detection



Physical attacks
- E.g., memory probing
- E.g., cold boot attack

Mobile Arm SoC

Integrated and Unified Memory in SoC

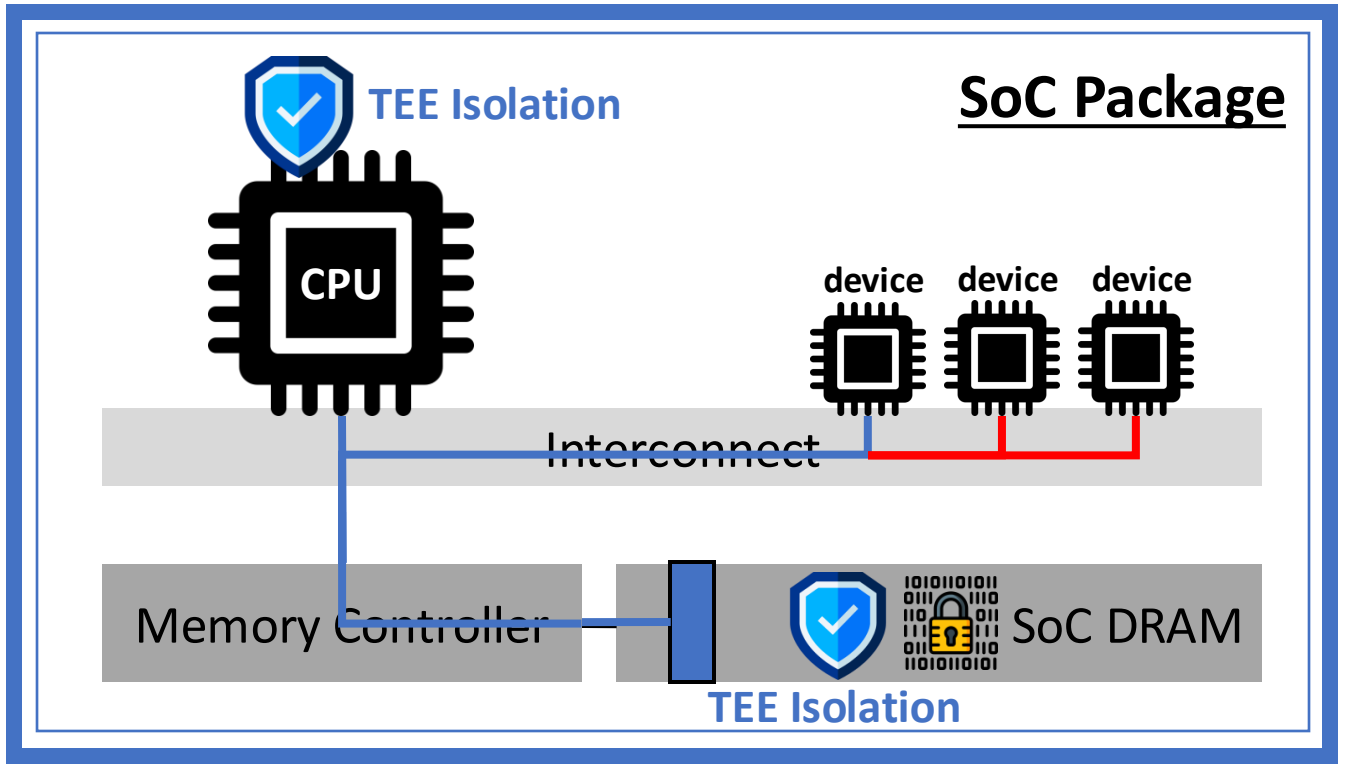


- Compact and integrated
- Advanced packaging technology
- Security features
  - E.g., tamper detection

Physical attacks
- E.g., memory probing
- E.g., cold boot attack

Mobile Arm SoC

Integrated and Unified Memory in SoC



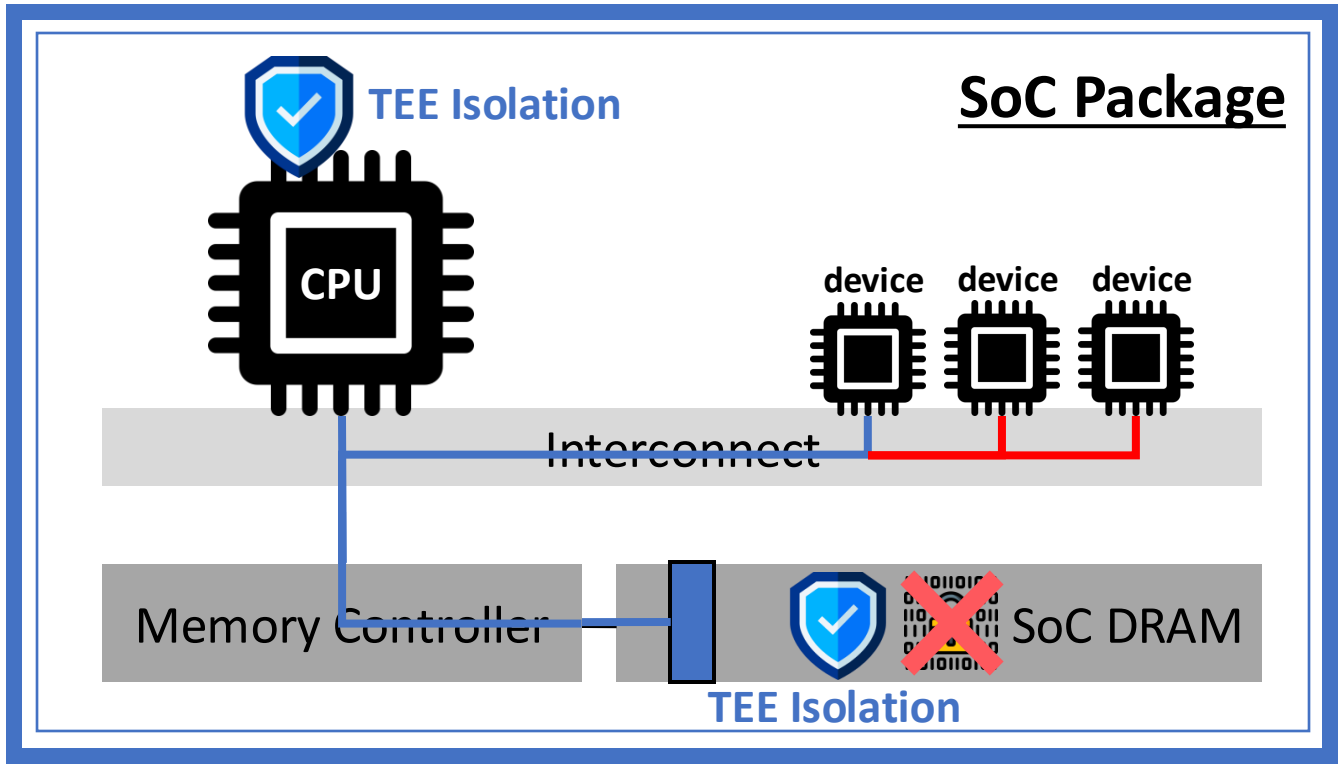
- Compact and integrated
- Advanced packaging technology
- Security features
  - E.g., tamper detection

Physical attacks

- E.g., memory probing
- E.g., cold boot attack

Mobile Arm SoC

Secure device I/O by isolation without memory encryption



- Compact and integrated
- Advanced packaging technology
- Security features
  - E.g., tamper detection

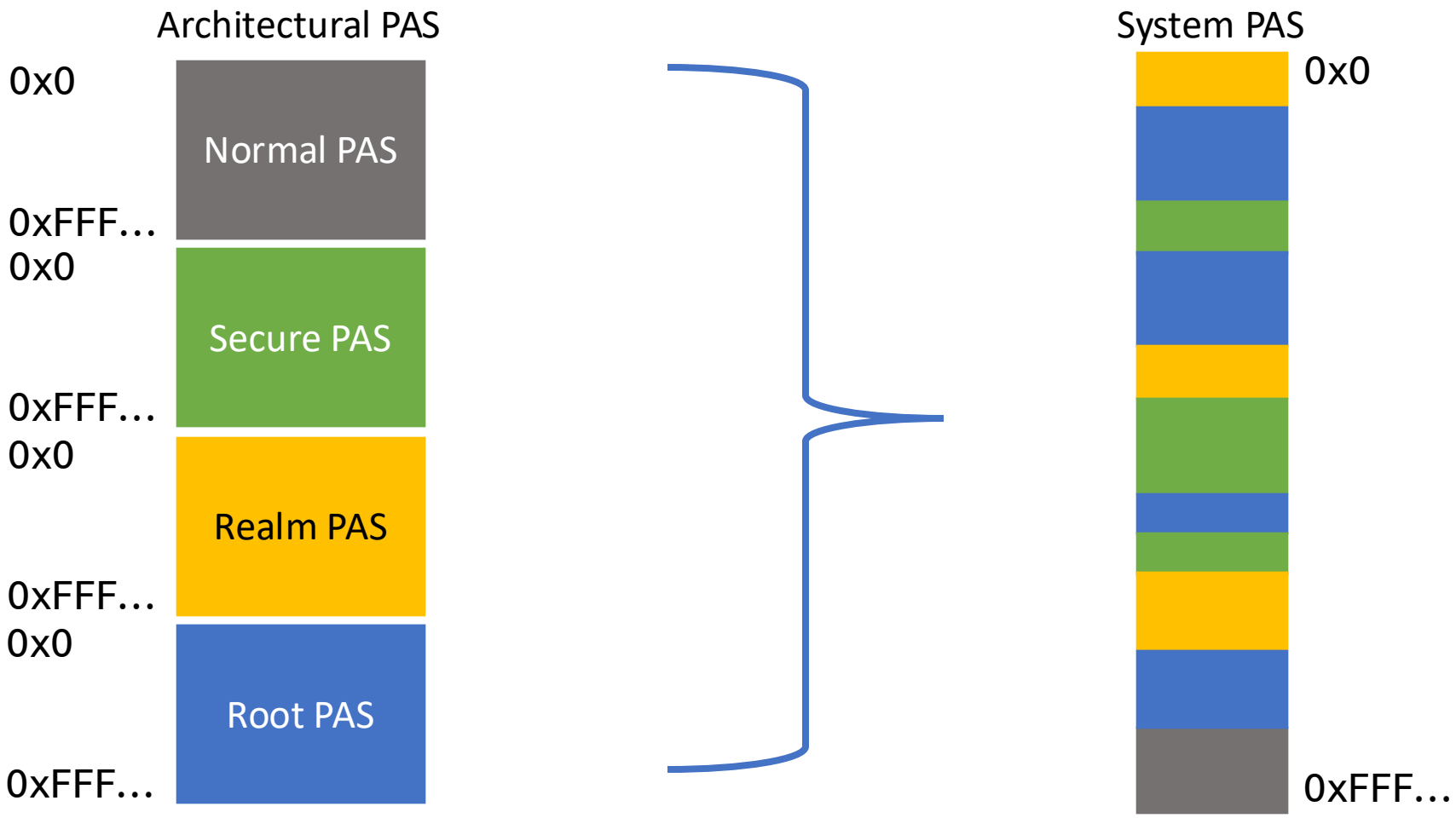
Physical attacks

- E.g., memory probing
- E.g., cold boot attack

Portal: Secure Device I/O For CCA on Mobile SoC

Arm Confidential Compute Architecture (CCA)

Physical Address Spaces (PAS)



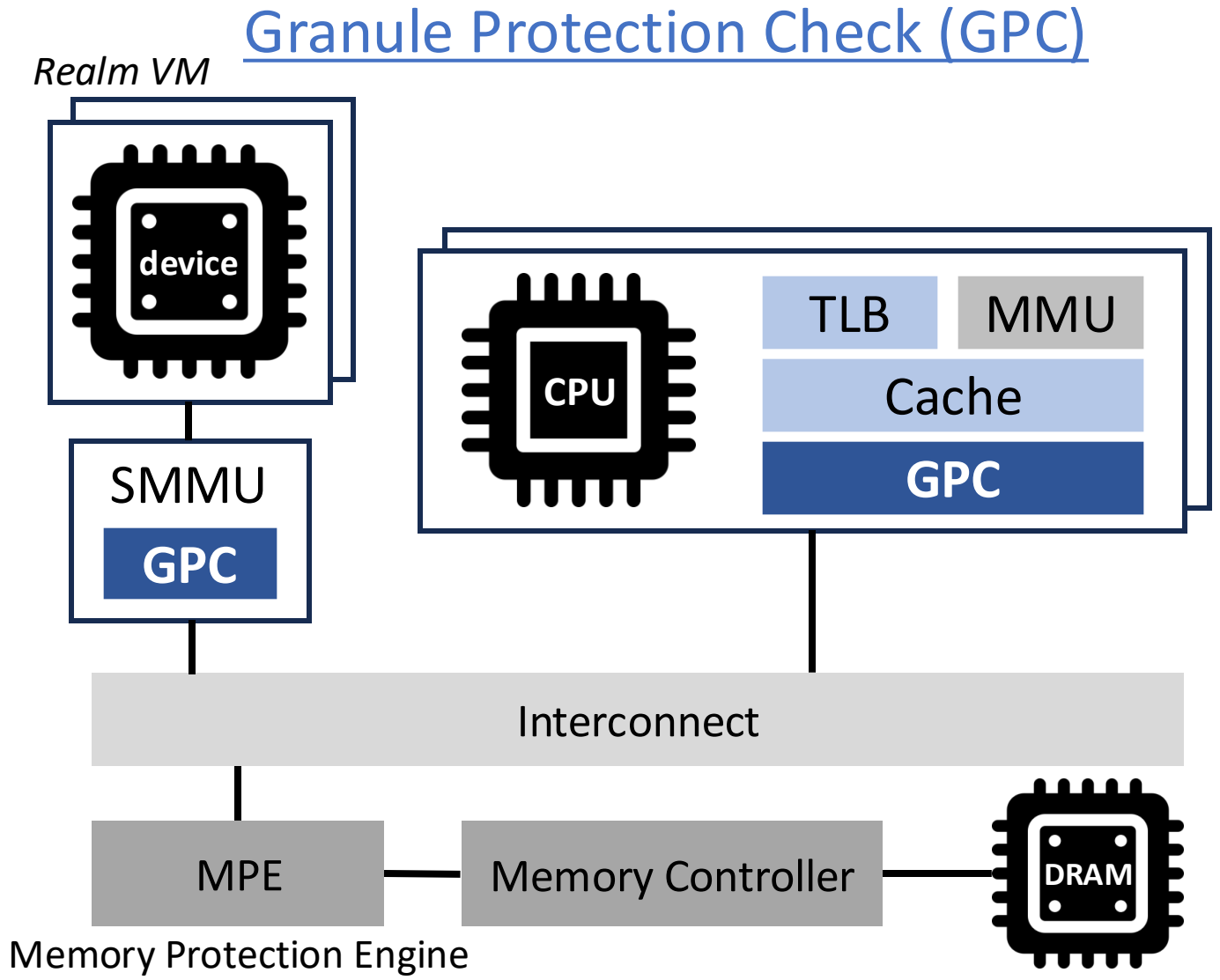
Arm Confidential Compute Architecture (CCA)

Granule Protection Check (GPC)

Security State	Normal PAS	Secure PAS	Realm PAS	Root PAS
Normal	✓	✗	✗	✗
Secure	✓	✓	✗	✗
Realm	✓	✗	✓	✗
Root	✓	✓	✓	✓

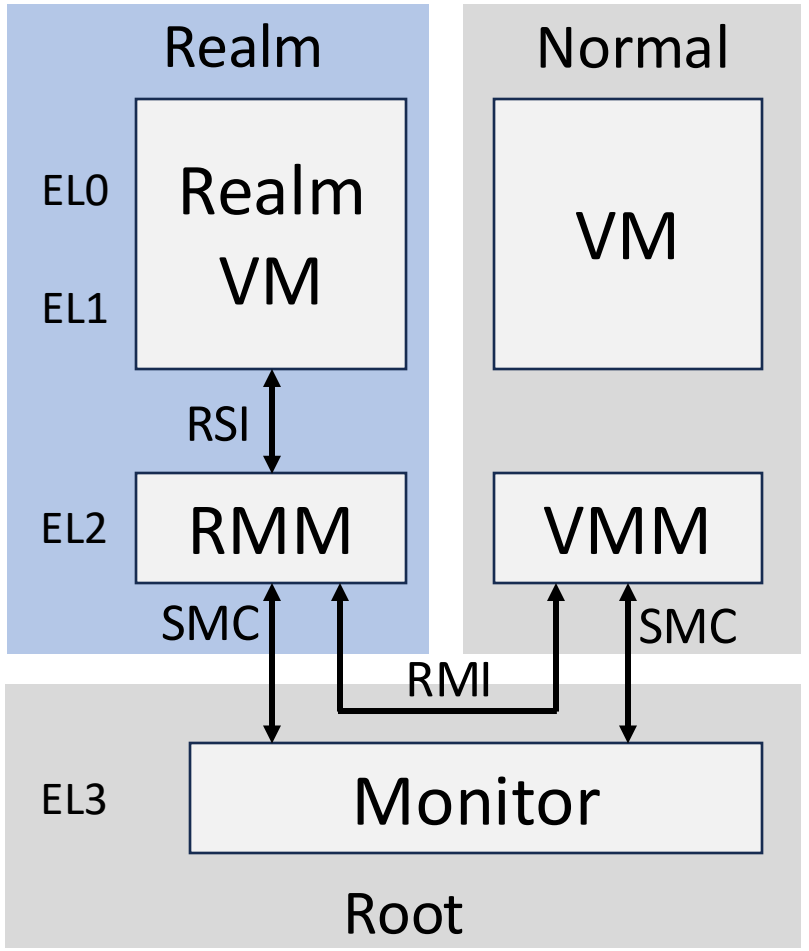
Granule Protection Table (GPT) stores PAS to world assignments and is managed in the Root world.

Arm Confidential Compute Architecture (CCA)



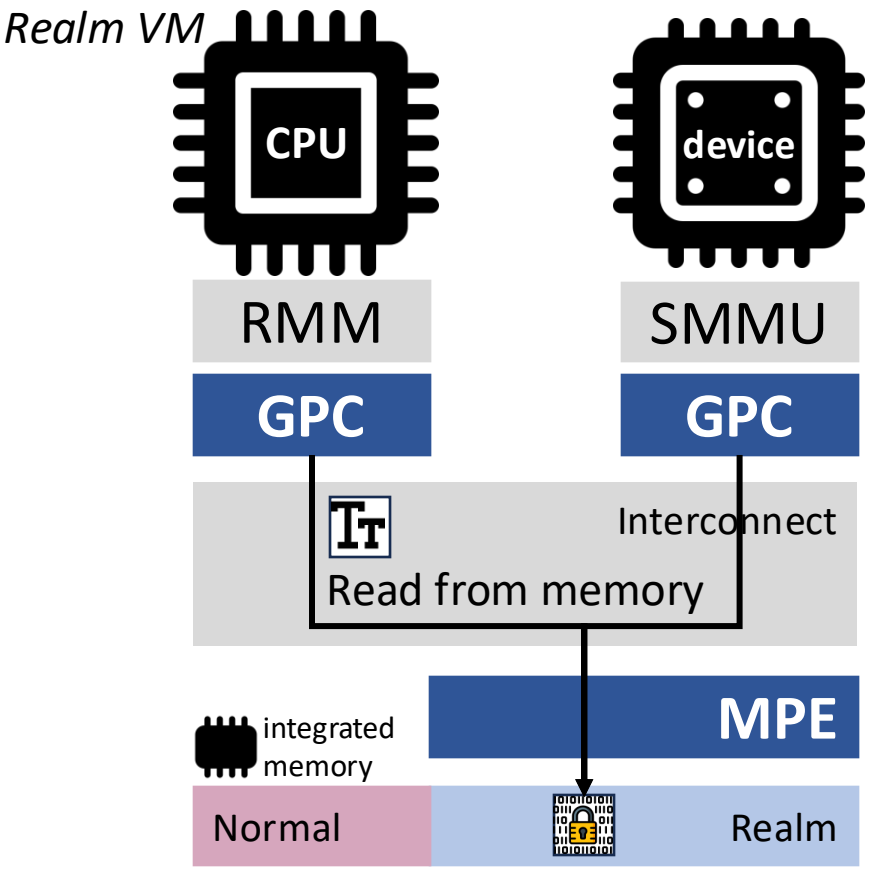
Arm Confidential Compute Architecture (CCA)

Interfaces between worlds

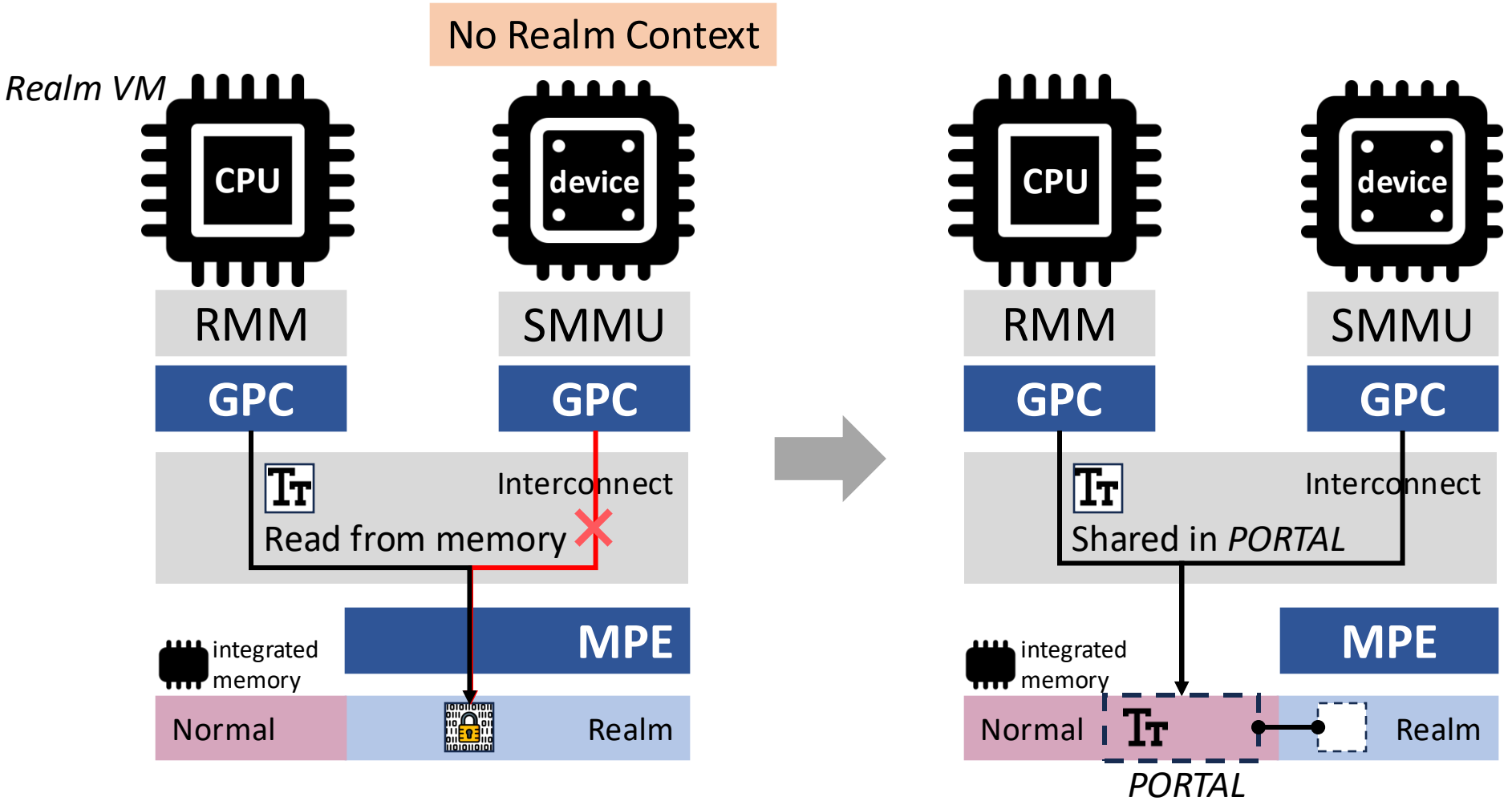


RSI - Realm service interface
RMI - Realm management interface

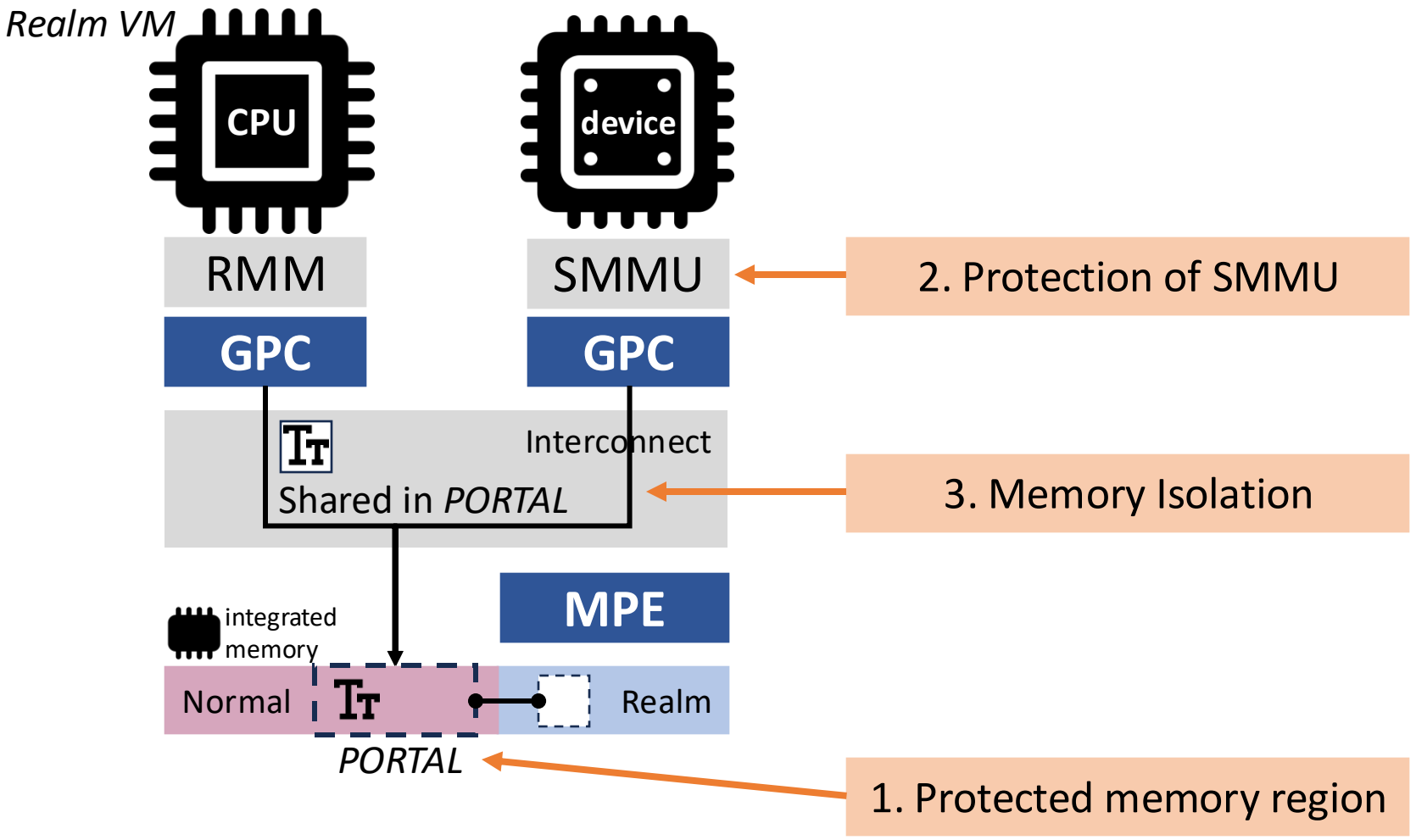
High Level Approach



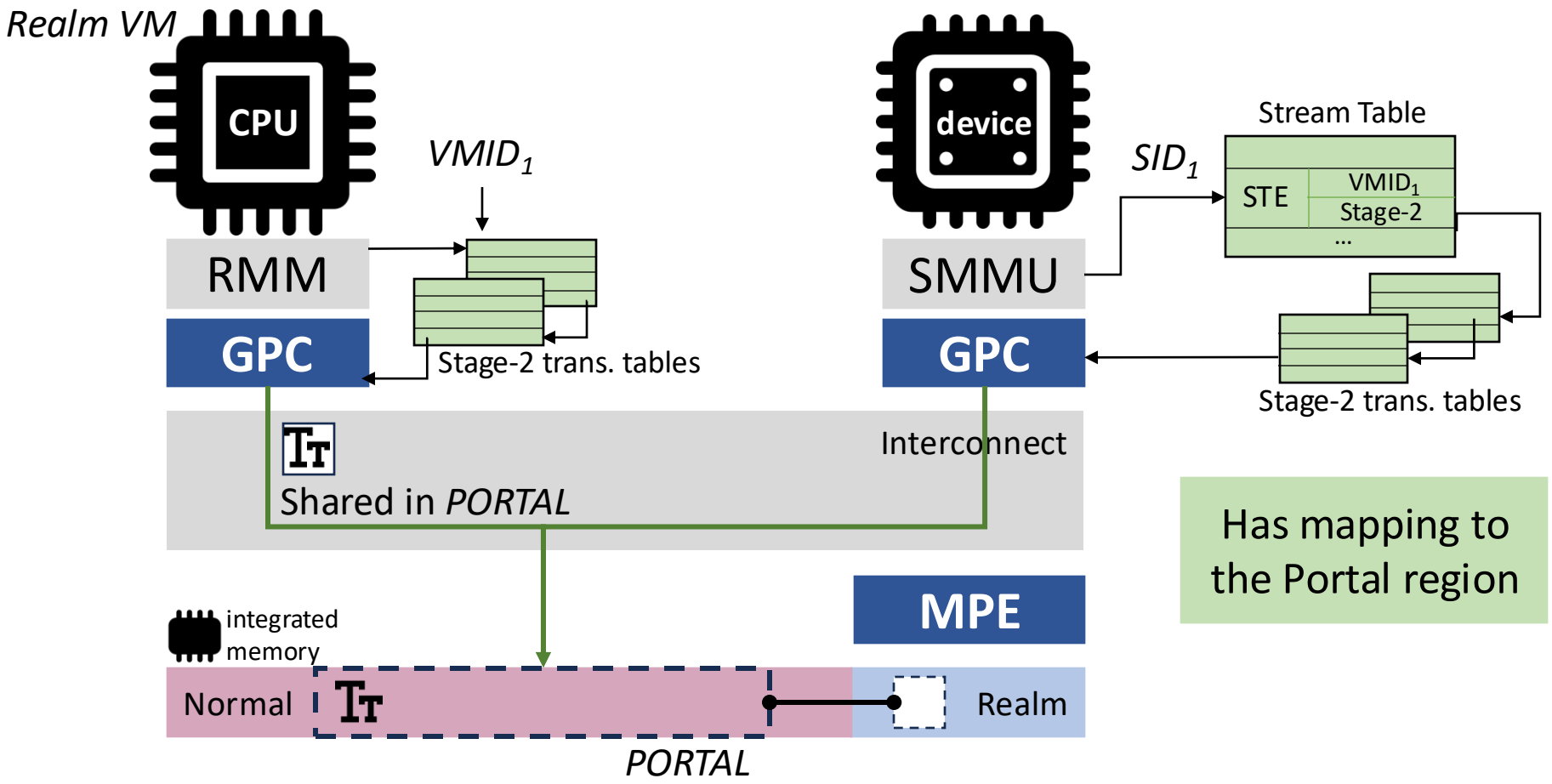
High Level Approach



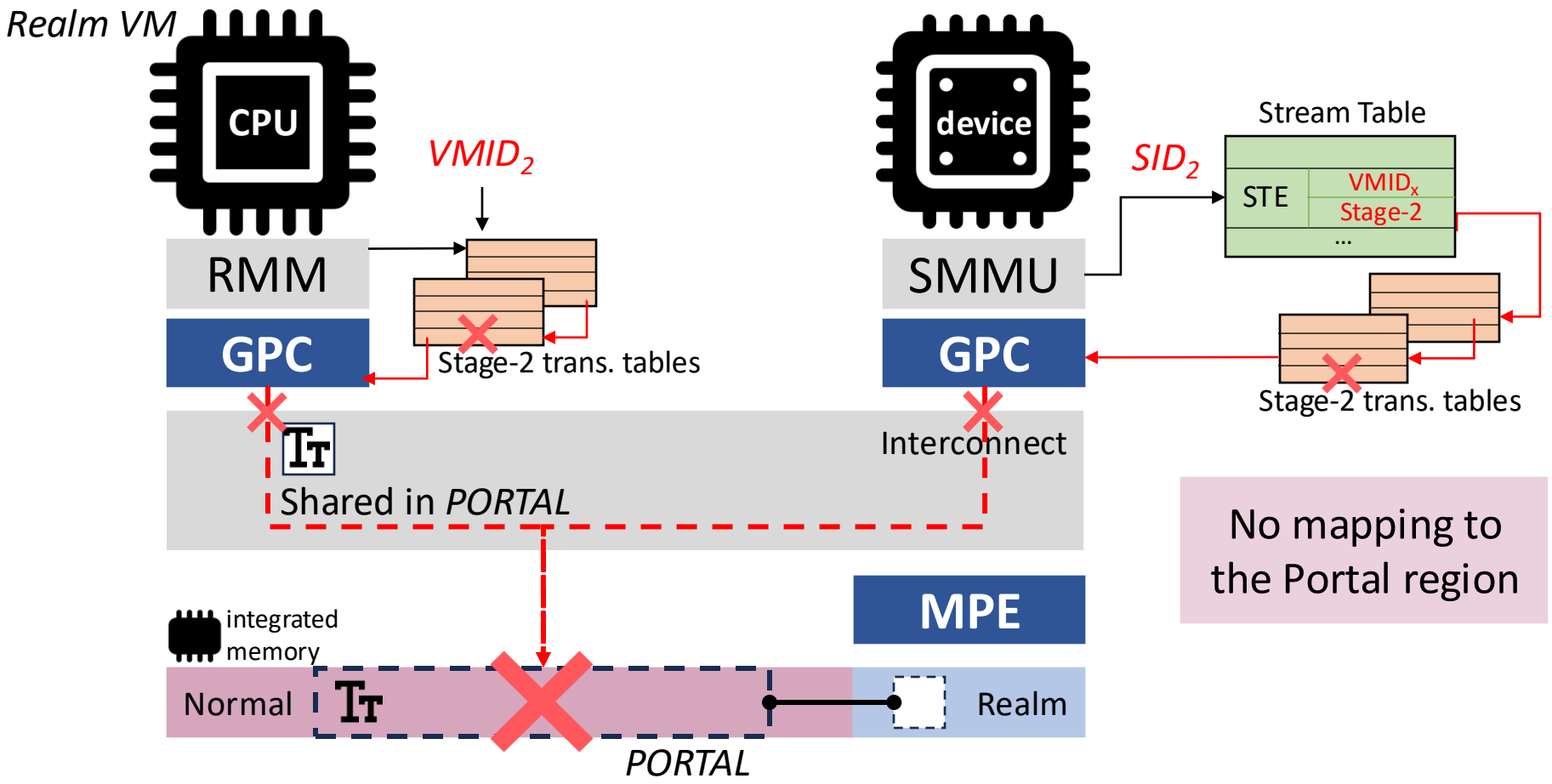
Requirements



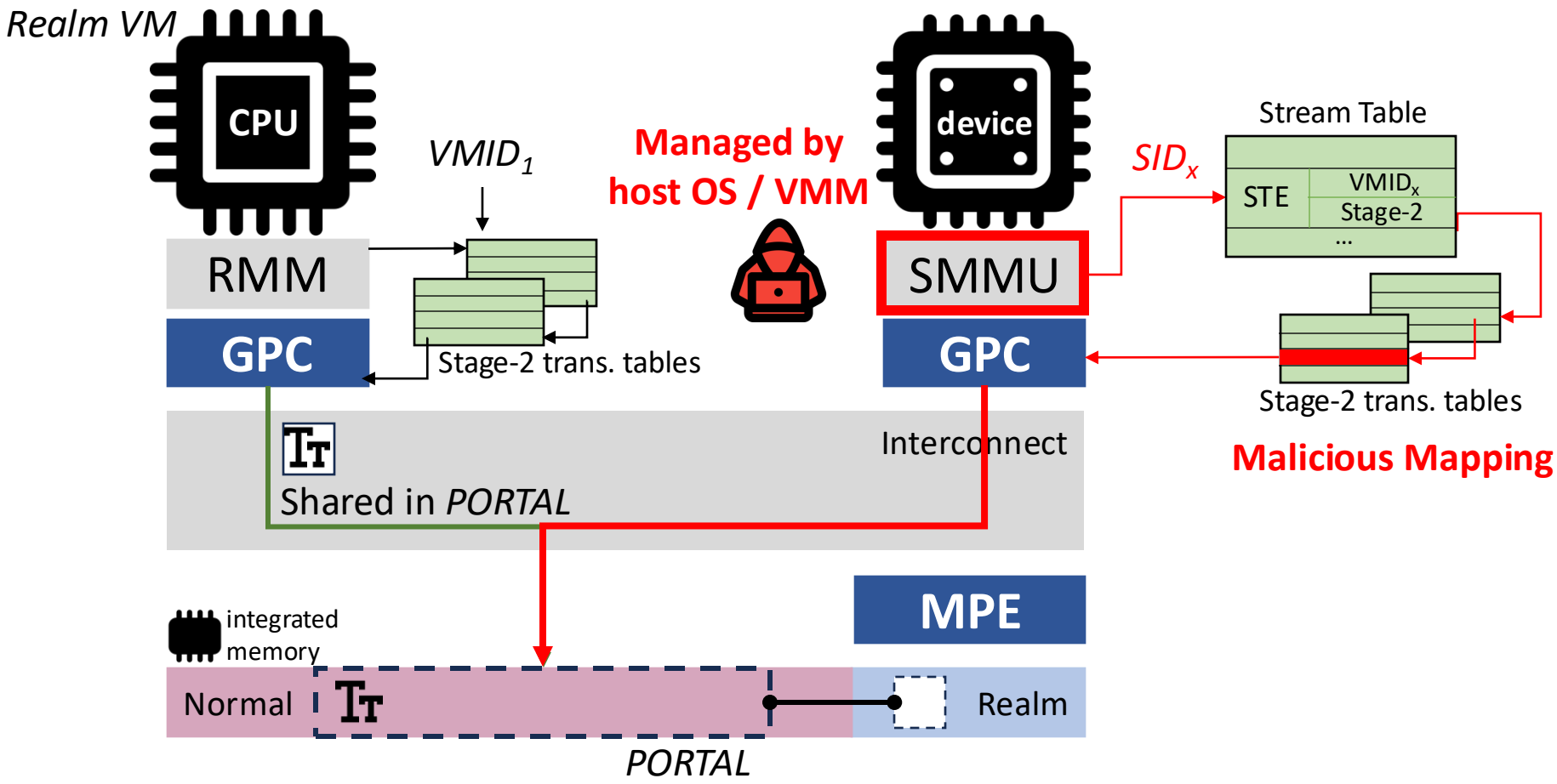
Protected Memory Region



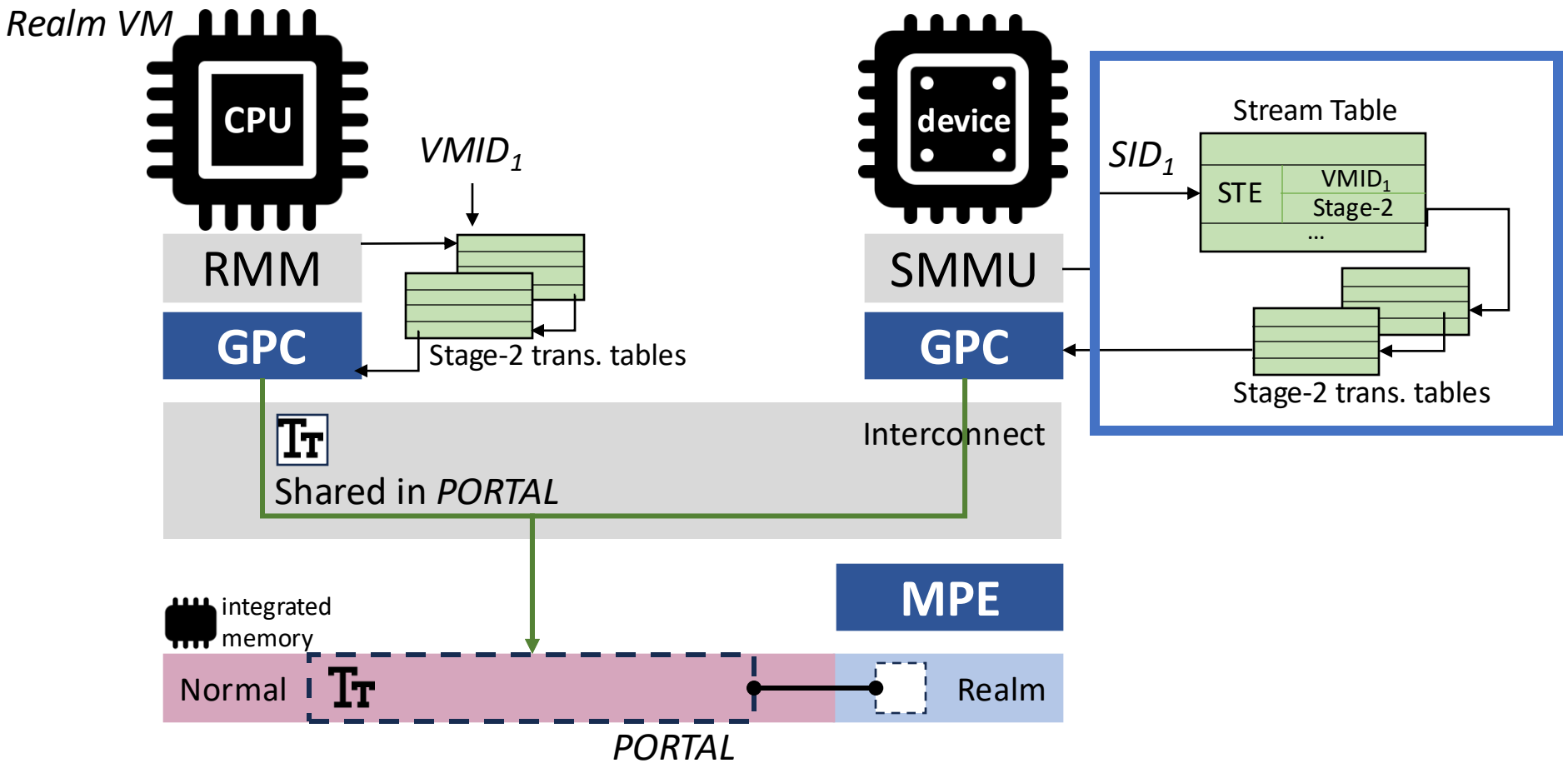
Protected Memory Region



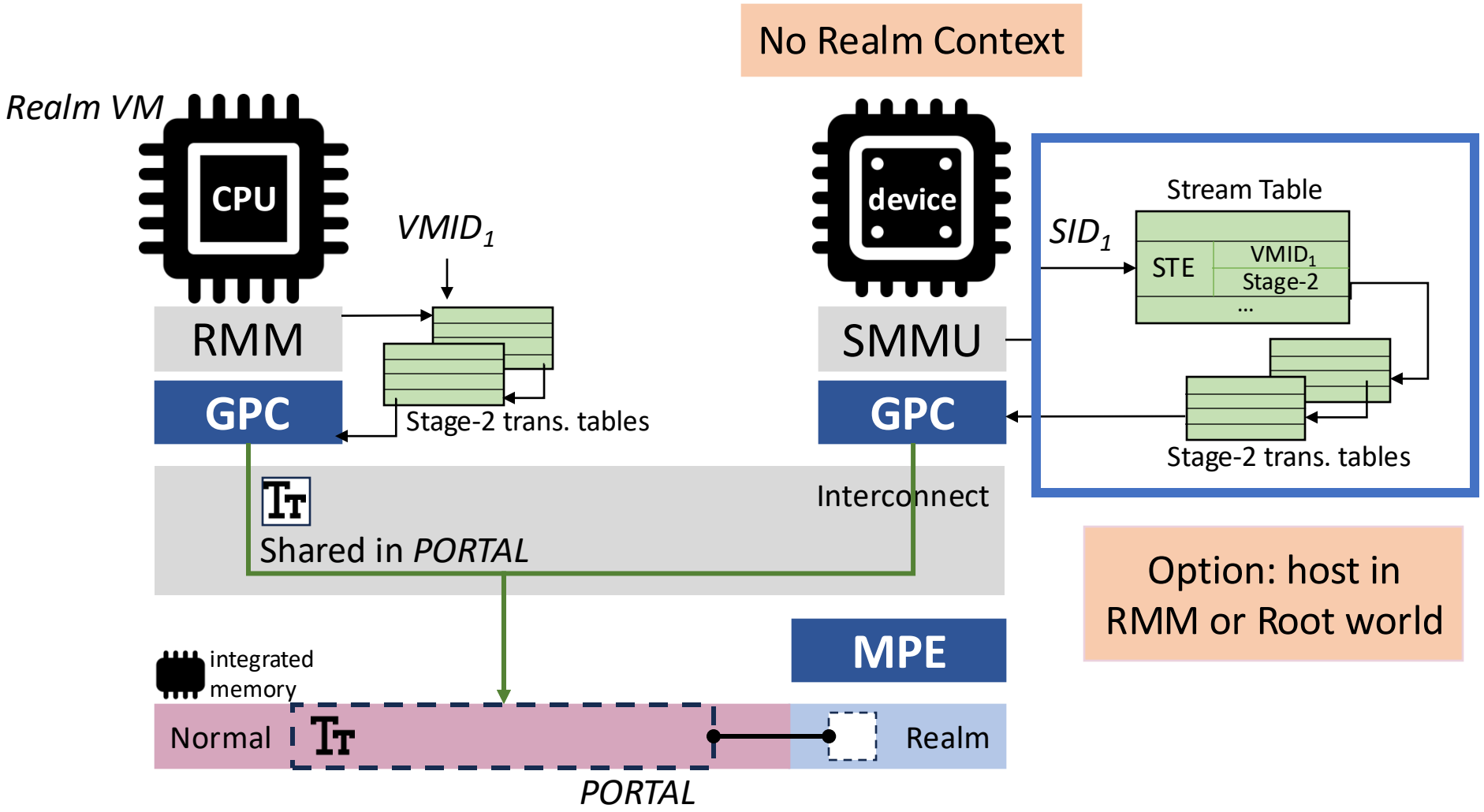
SMMU Managed by Untrusted Host OS / VMM



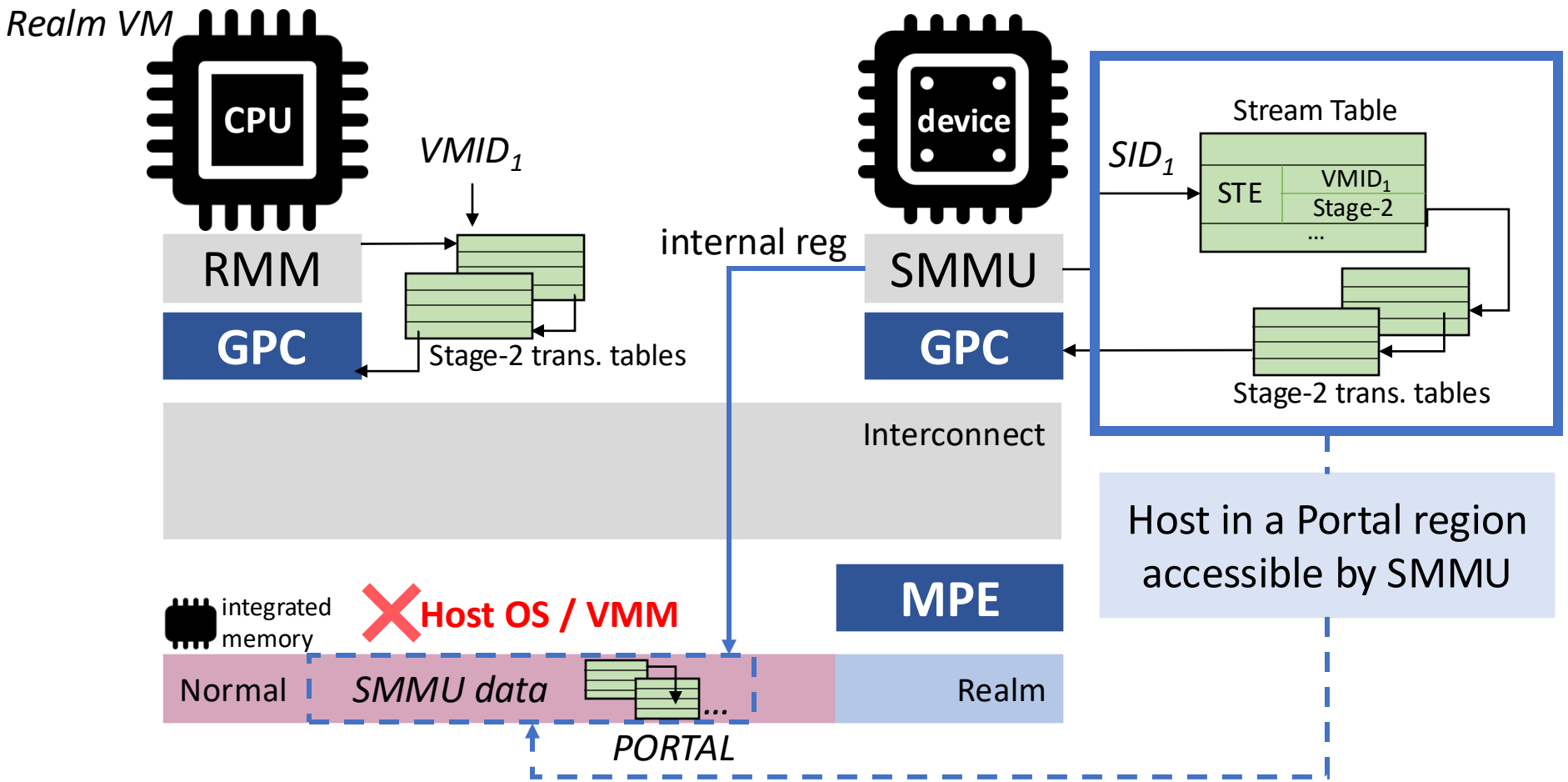
Protection of SMMU Data Structures



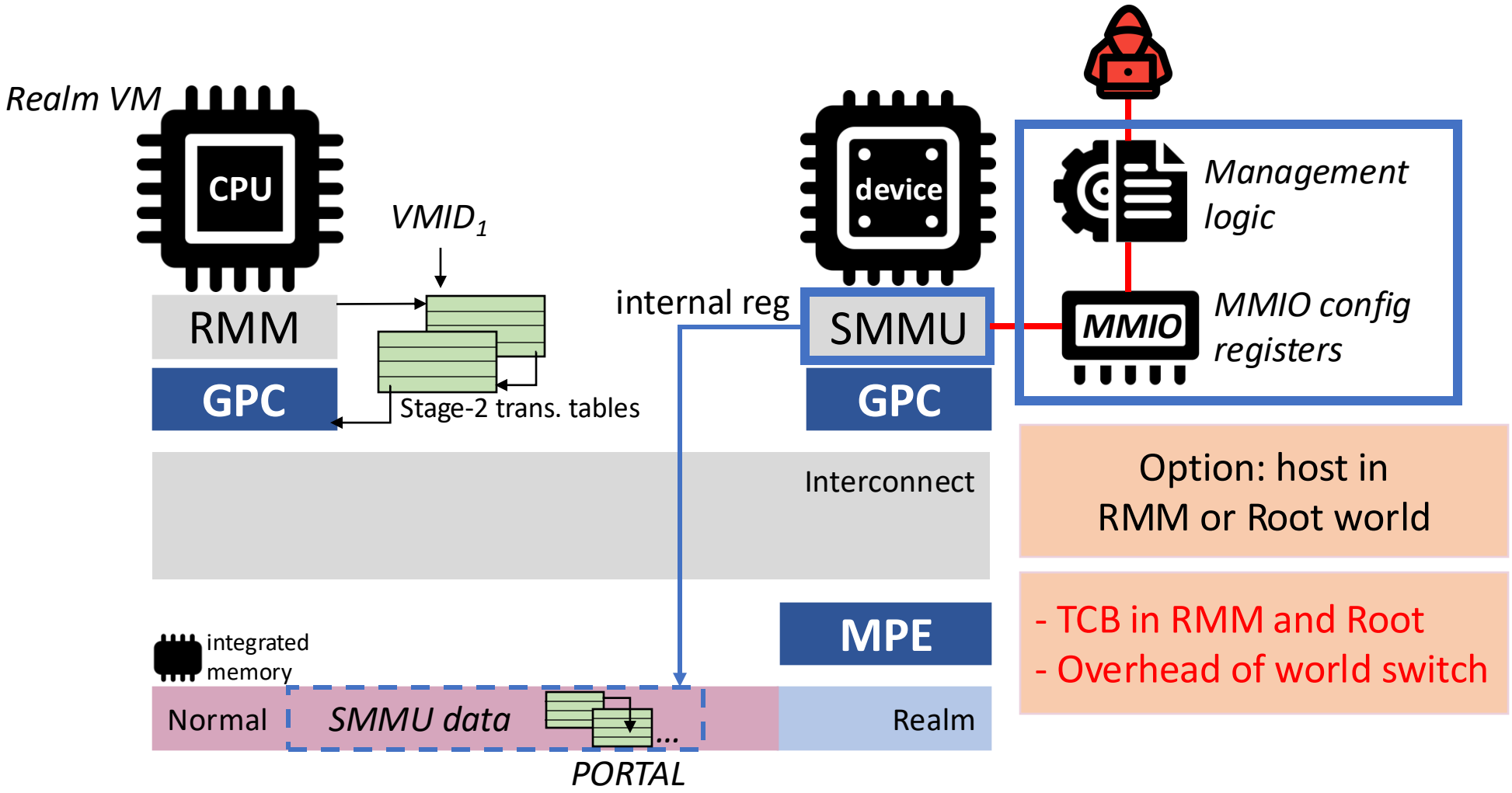
Protection of SMMU Data Structures



Protection of SMMU Data Structures



Protection of SMMU Management

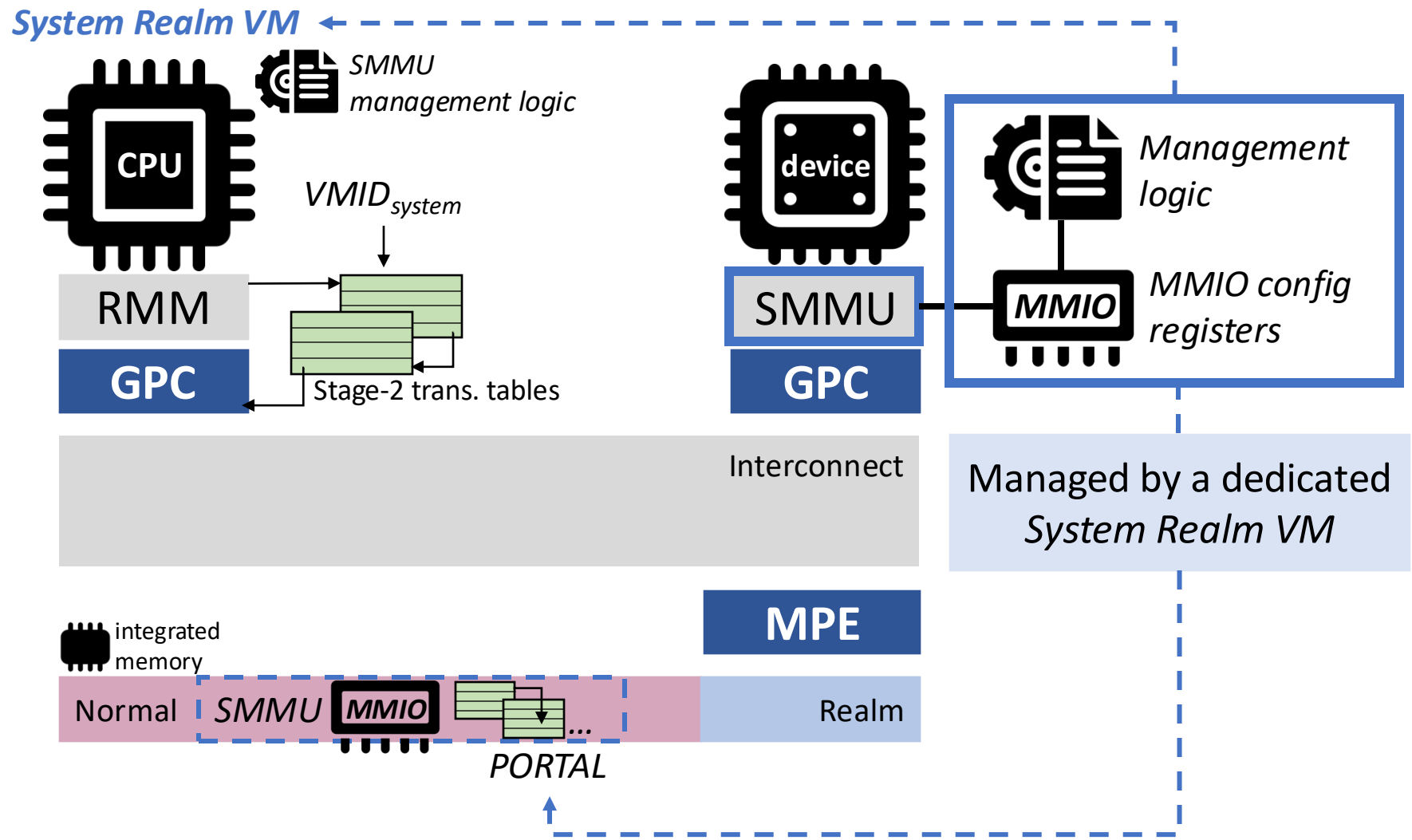


Option: host in RMM or Root world

- TCB in RMM and Root
- Overhead of world switch

Protection of SMMU Management

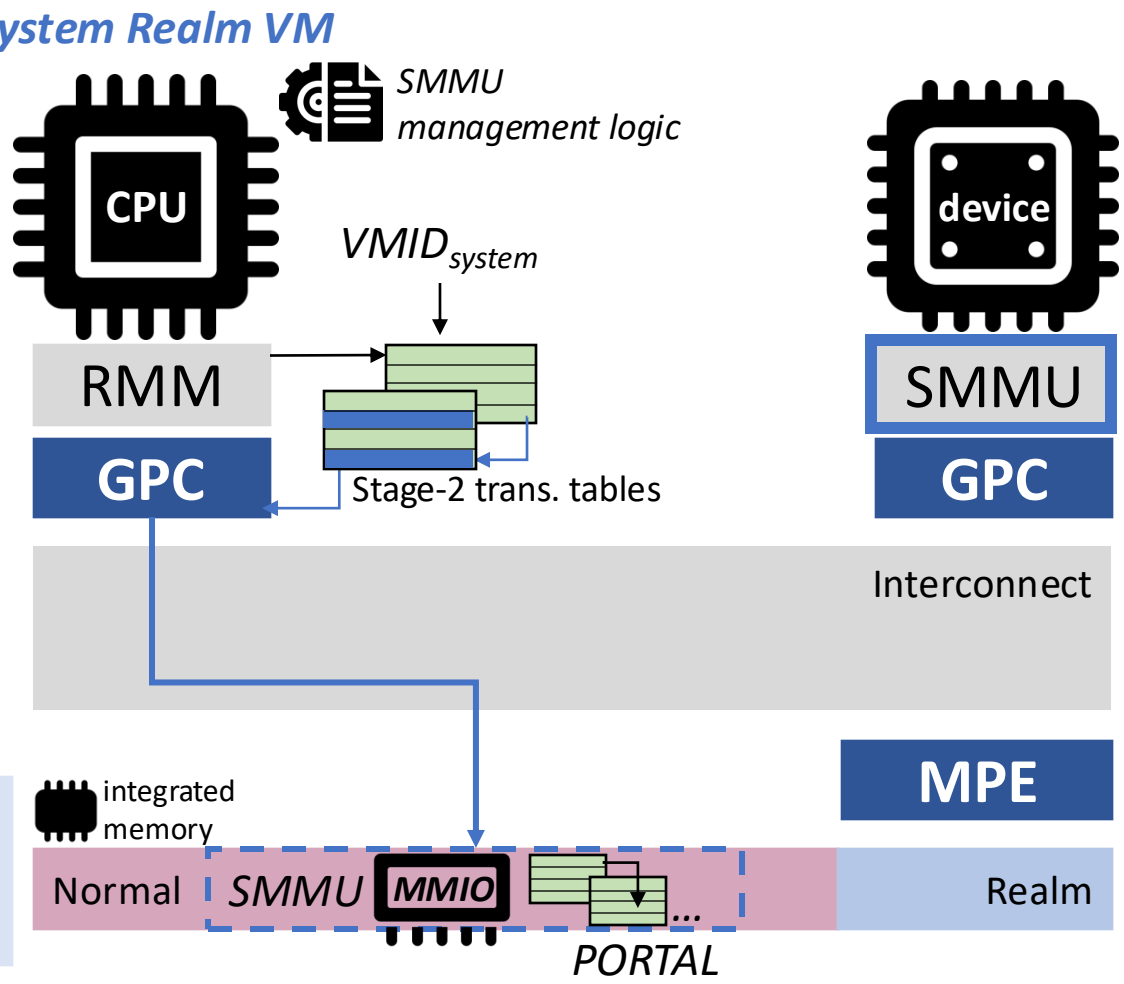
Manage SMMU *on behalf of* host and other Realms



Protection of SMMU Management

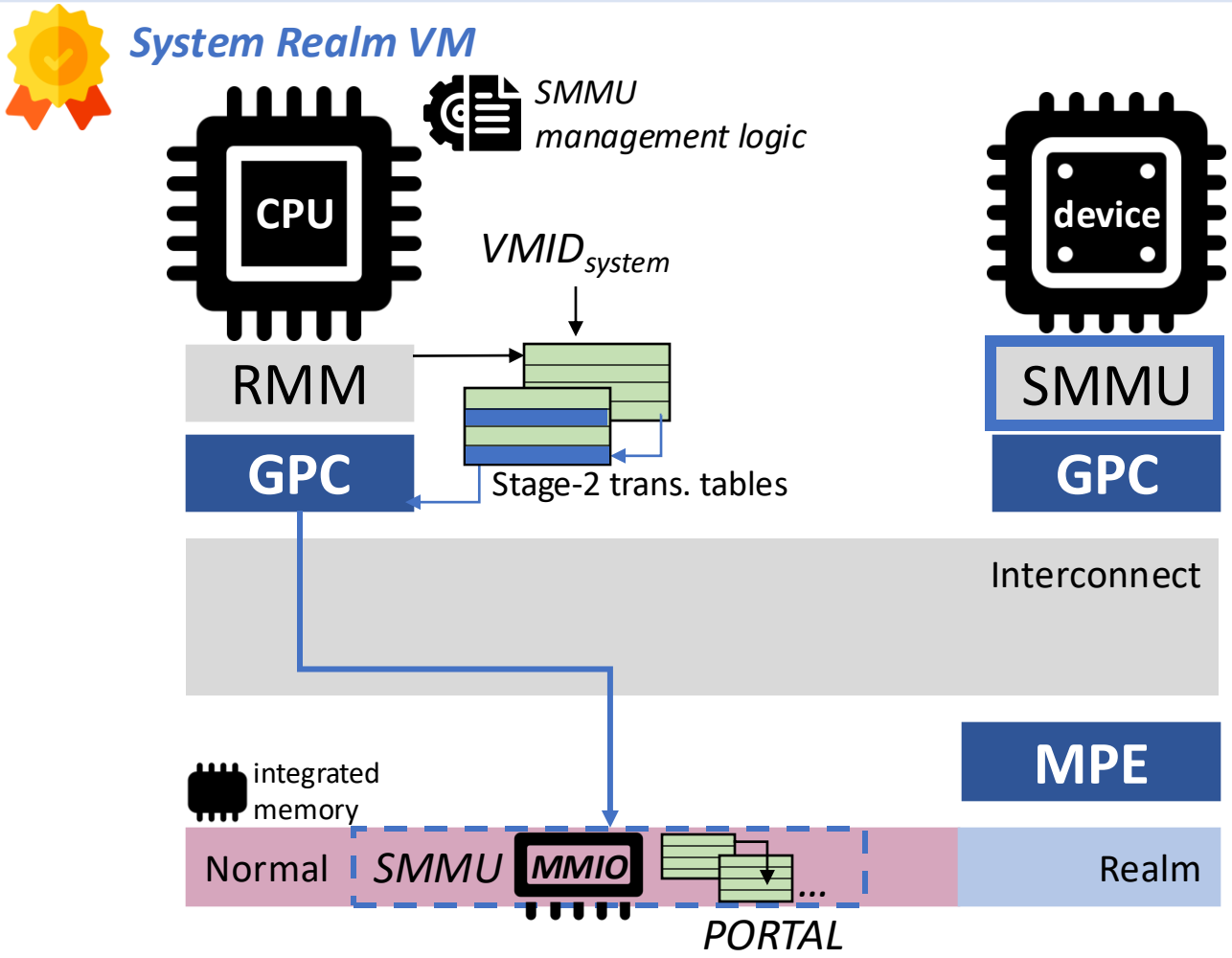
Owns the Portal region of SMMU data structures

Exclusive access to the MMIO configuration registers



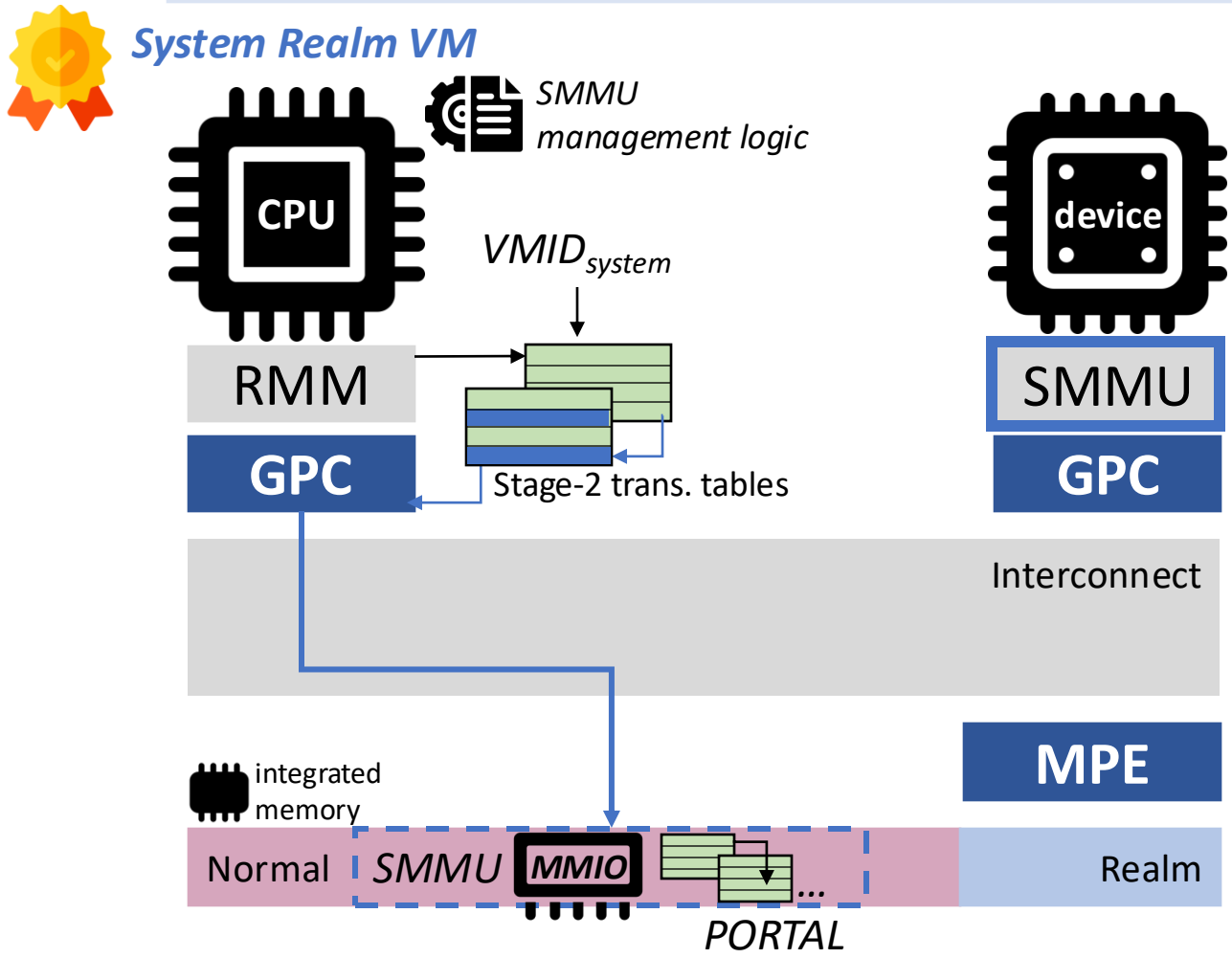
Protection of SMMU Management

System Realm VM should be implemented and distributed by trusted vendors



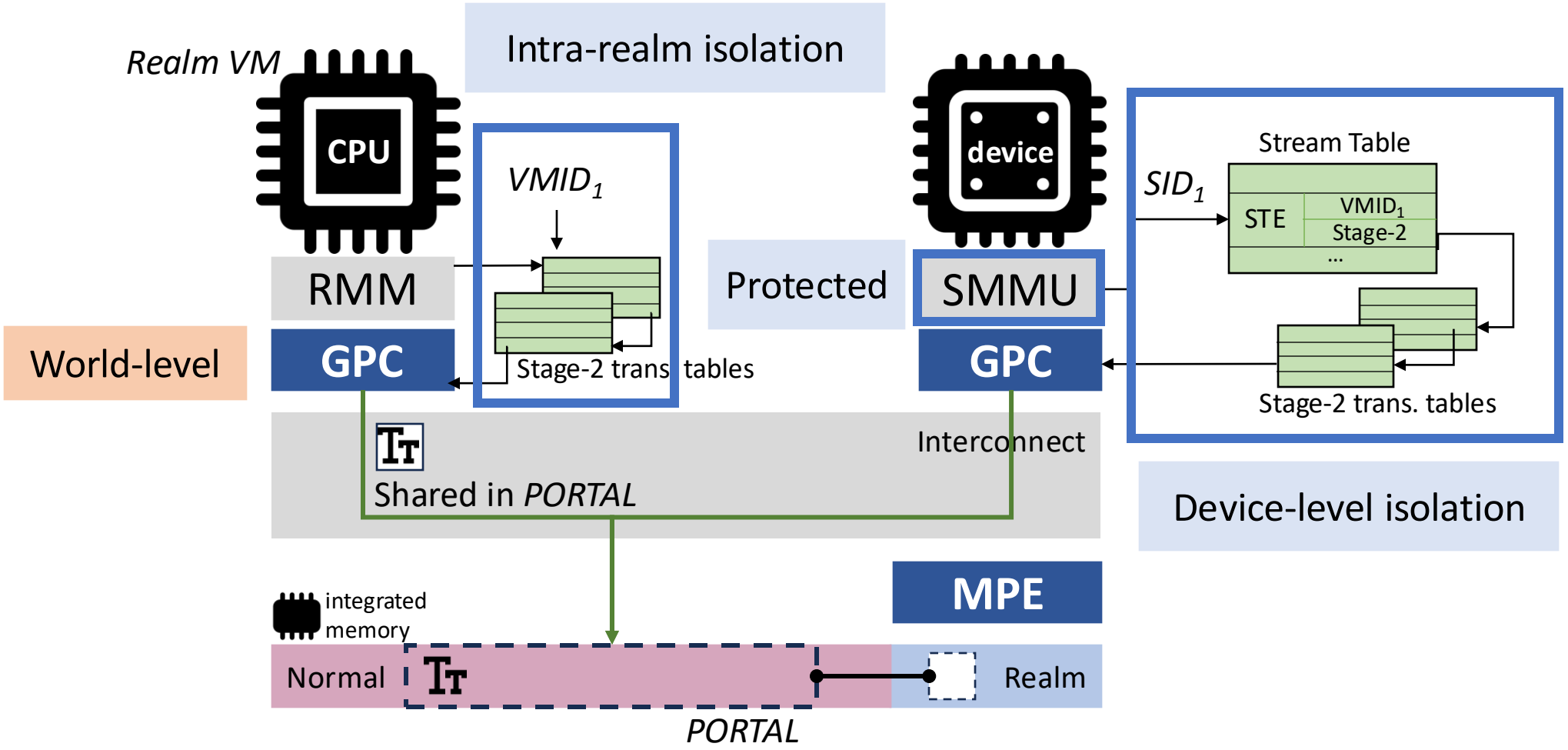
System Realm as a Realm VM

System Realm VM should be implemented and distributed by trusted vendors

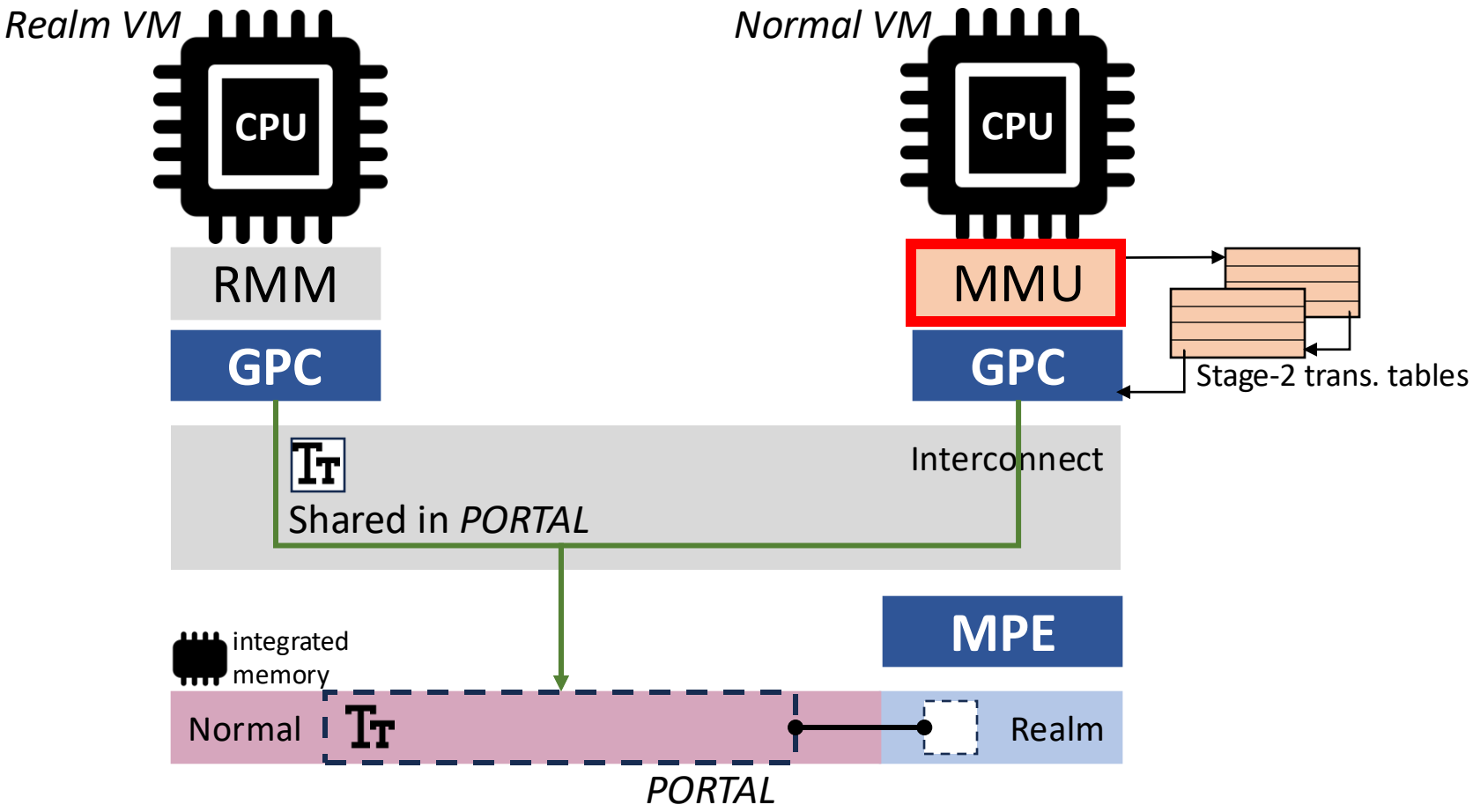


- Verify through attestation
- Isolated from EL2 and EL3
- System Realm interface to reduce overhead

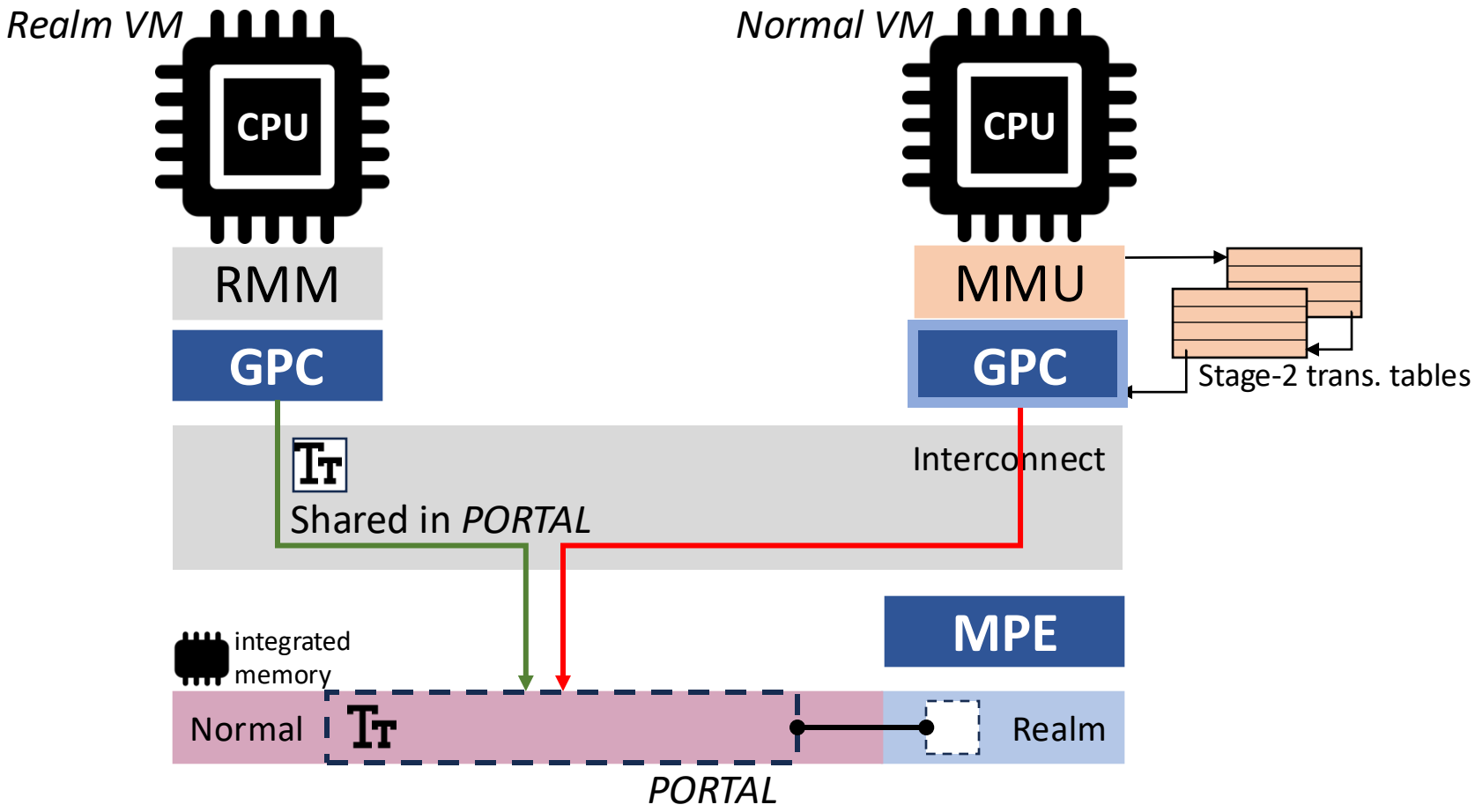
Intra-Realm and Device-Level Isolation



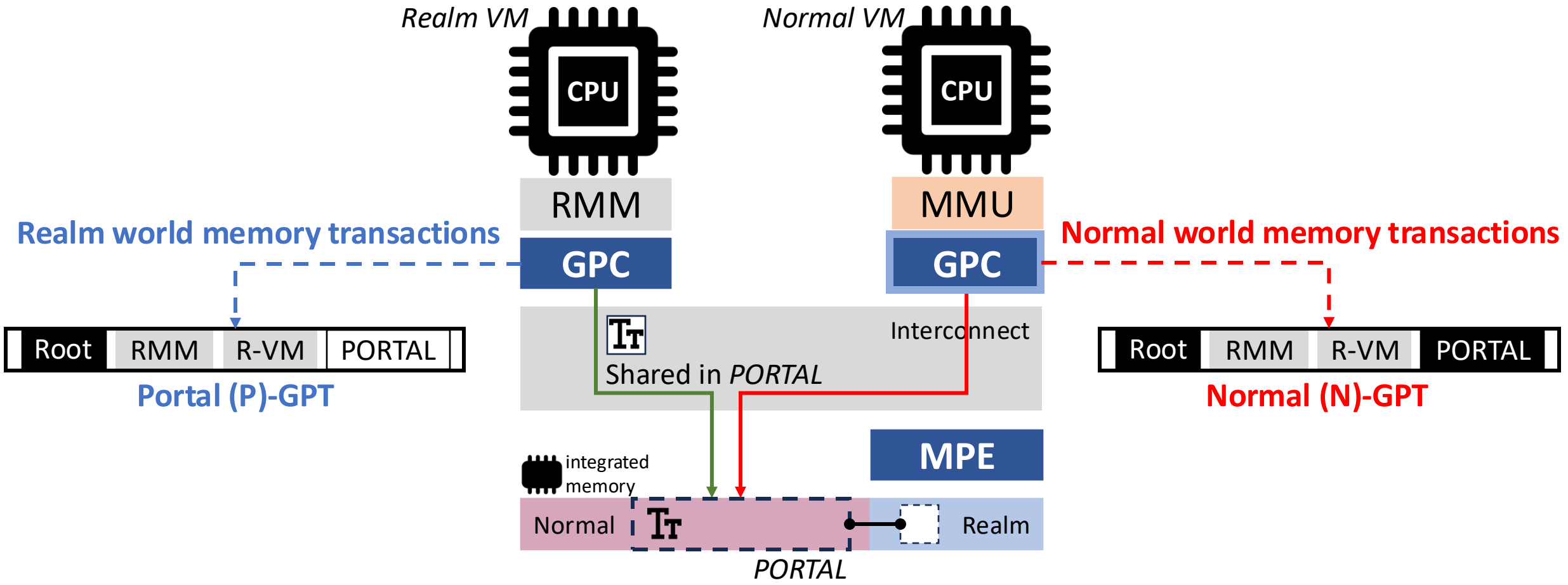
Isolating Authorized and Unauthorized Entities



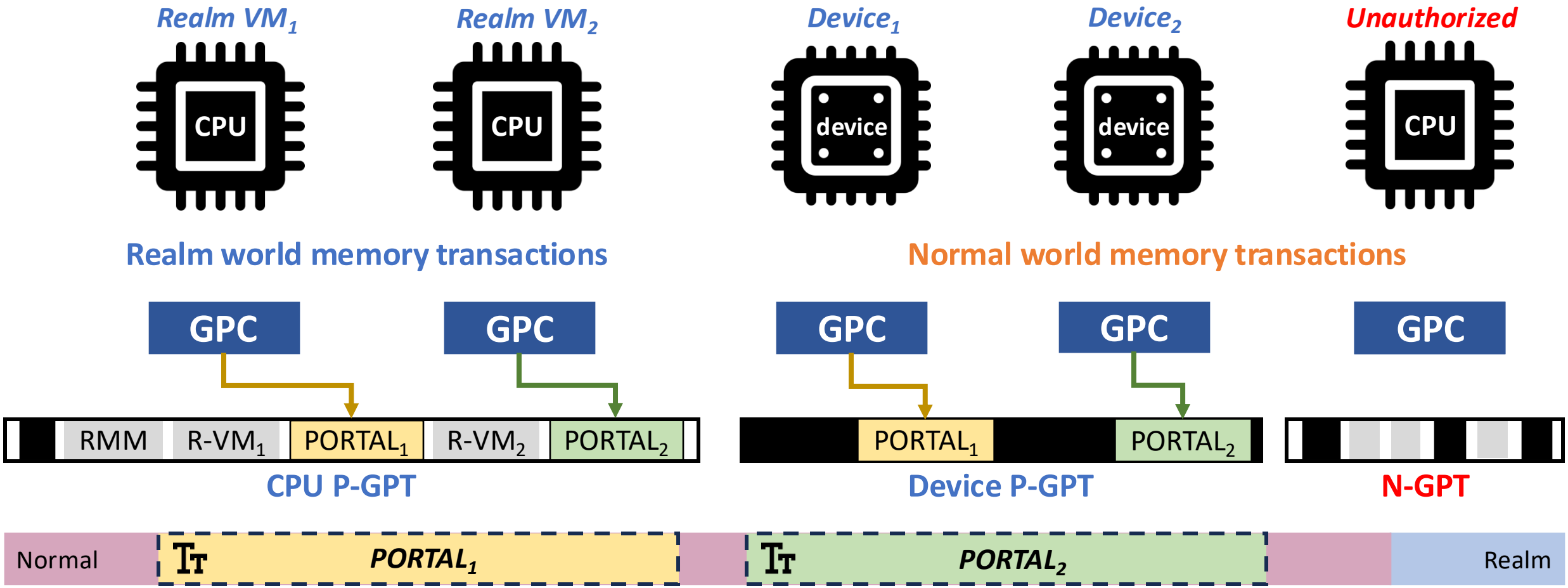
Isolating Authorized and Unauthorized Entities



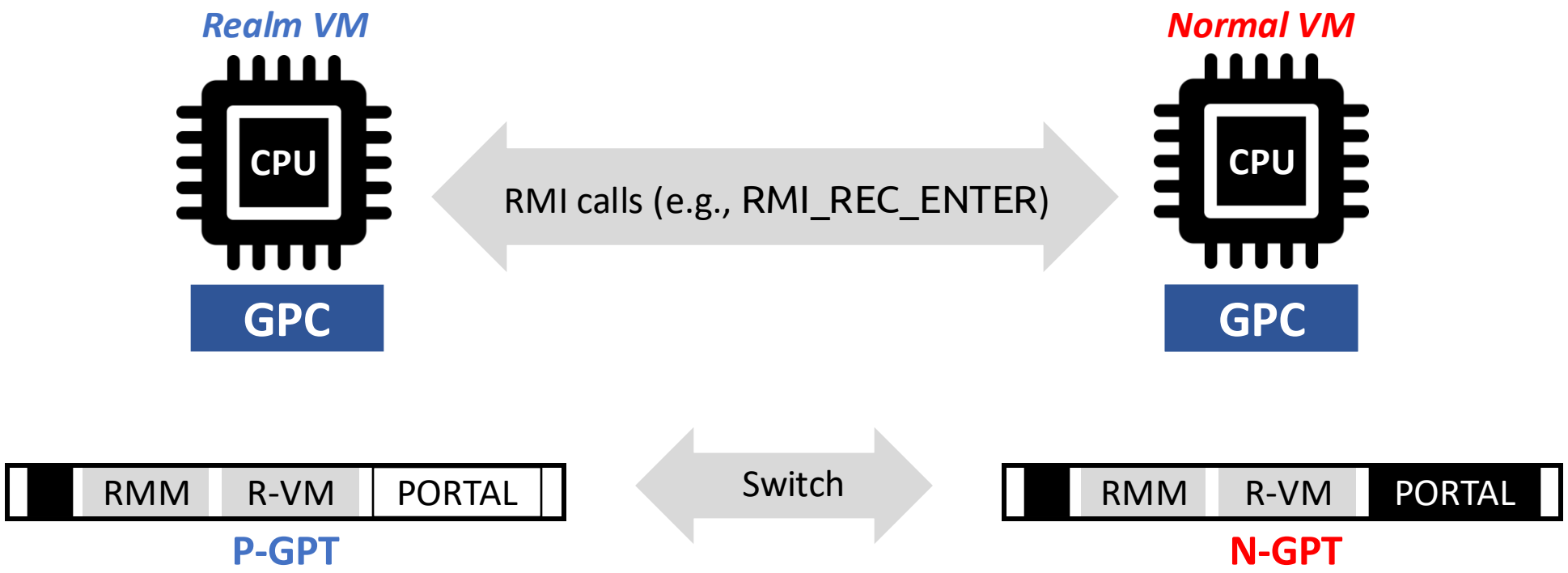
Isolating Authorized and Unauthorized Entities



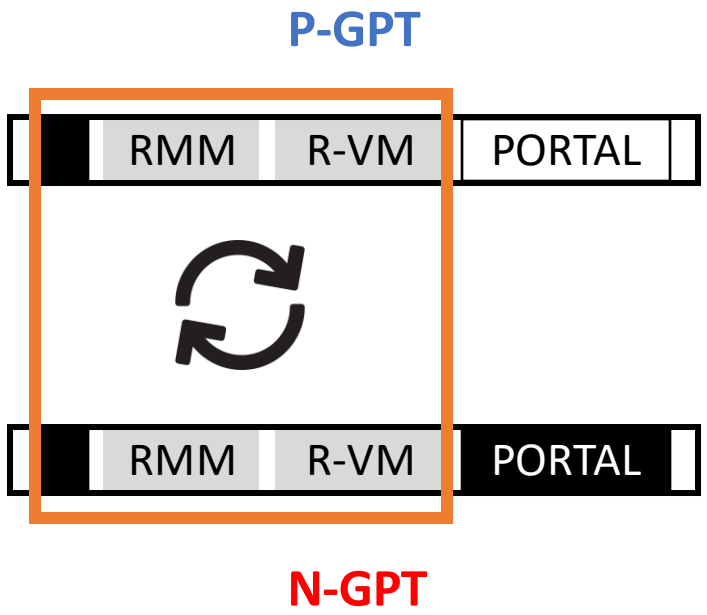
Isolating Authorized and Unauthorized Entities



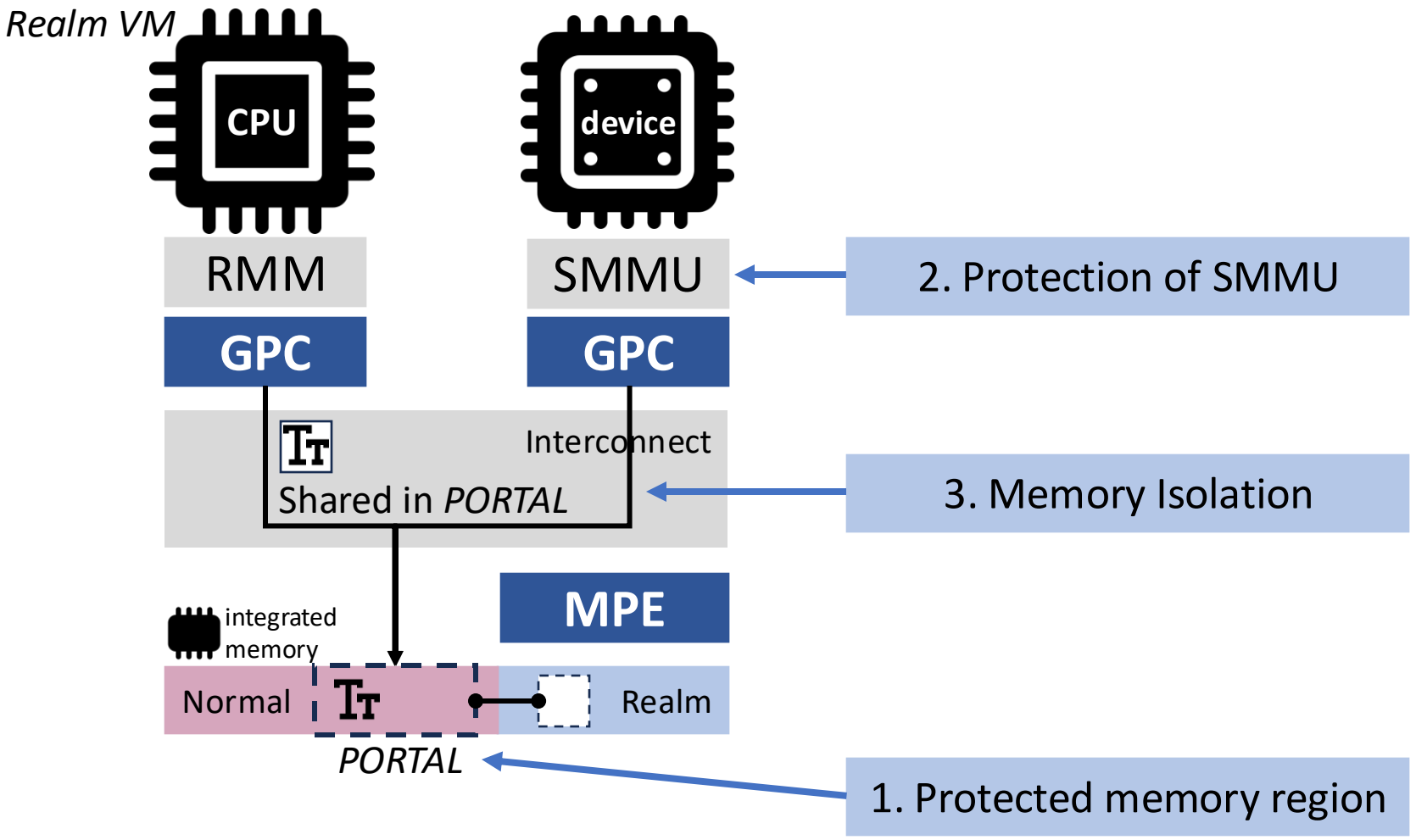
Switching between P-GPT and N-GPT



Synchronizing P-GPT and N-GPT



Requirements

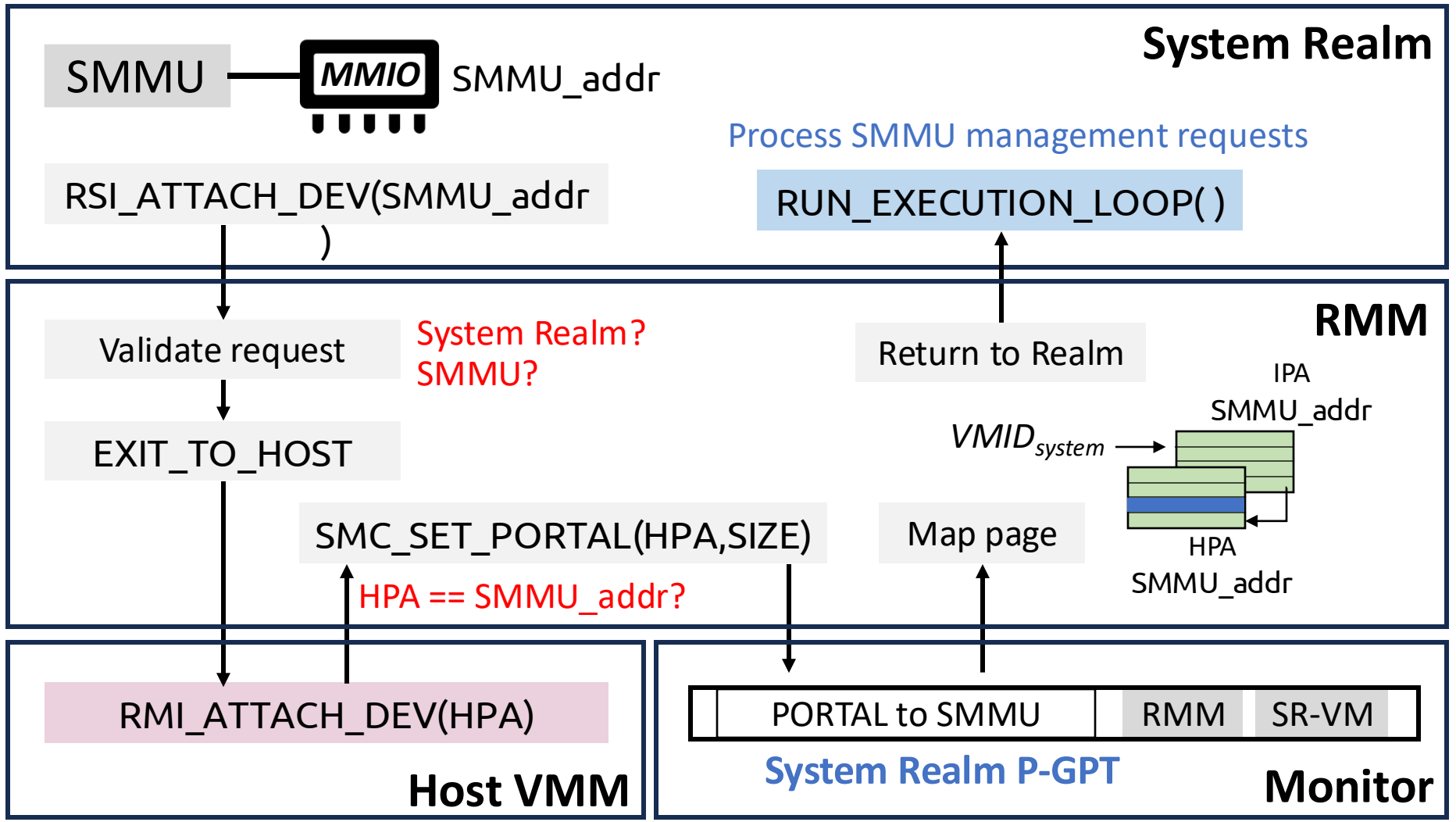


System Realm for Exclusive SMMU Management

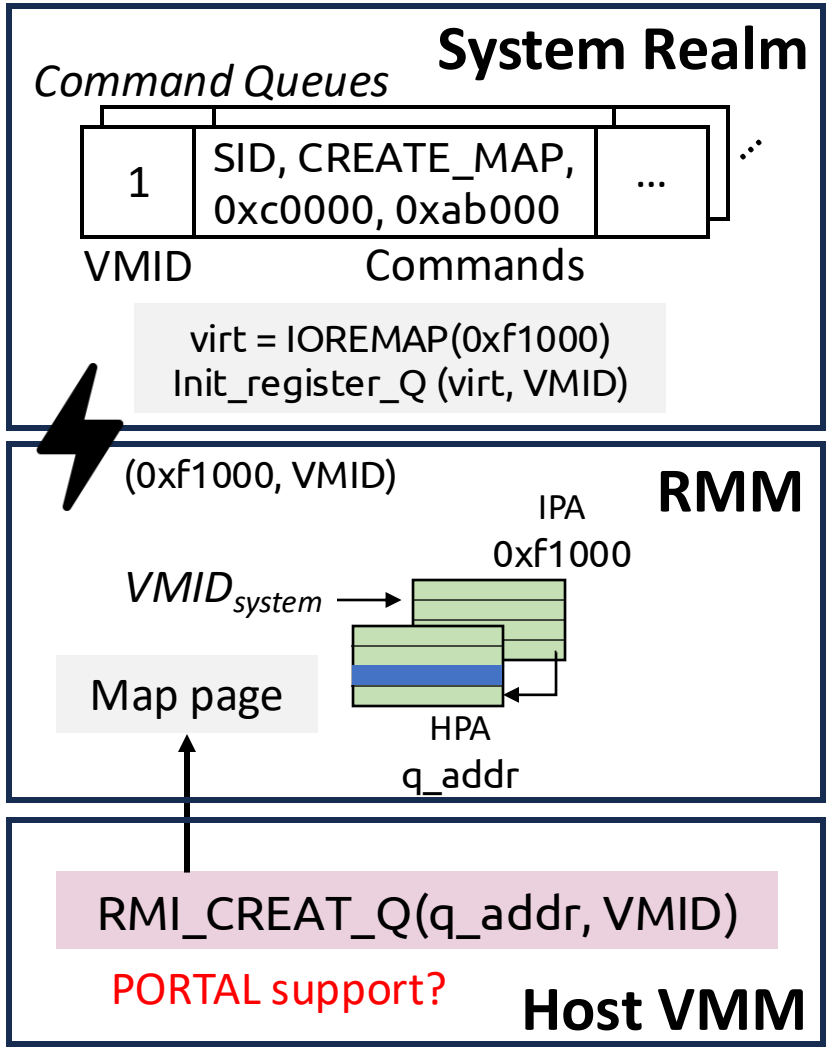
System Realm Interfaces

API	Description
RMI_REC_ENTER*	initiate the execution of a Realm VM. PORTAL updates P-GPT and N-GPT. PORTAL check the device attachment / detachment.
RSI_ATTACH_DEV	allocate a device from a Realm VM.
RSI_DEV_MNG	attach and detach a device from Realm world.
RSI_SET_PORTAL	request a PORTAL region for DMA.
RMI_ATTACH_DEV	allocate device memory from Normal world to Realm world.
RMI_CREATE_Q	create a command queue for a newly instantiated Realm VM.
SMC_SET_PORTAL	delegate Realm memory to a PORTAL region. add stage-2 translation for the SMMU.

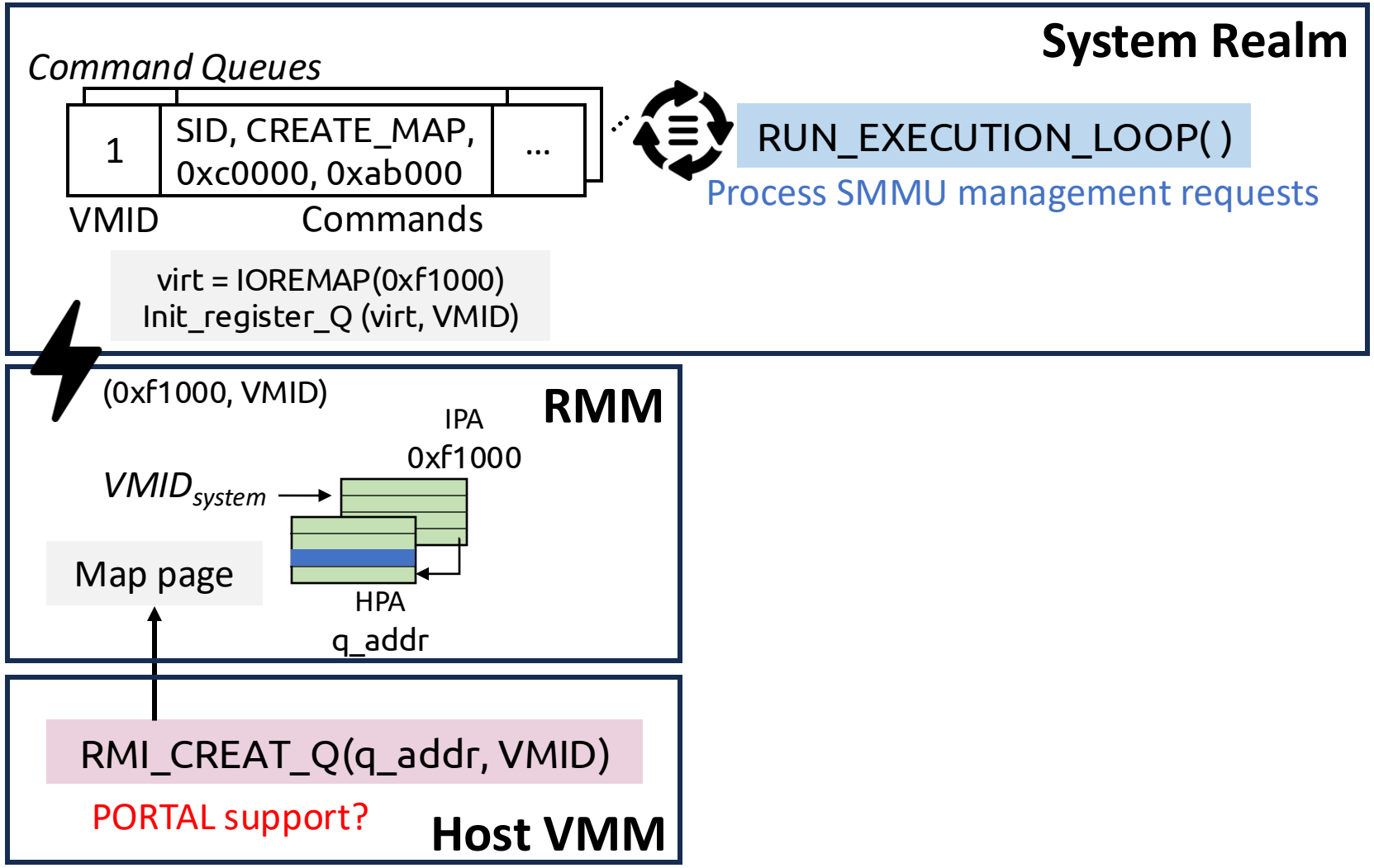
System Realm Initialization



System Realm Command Queue

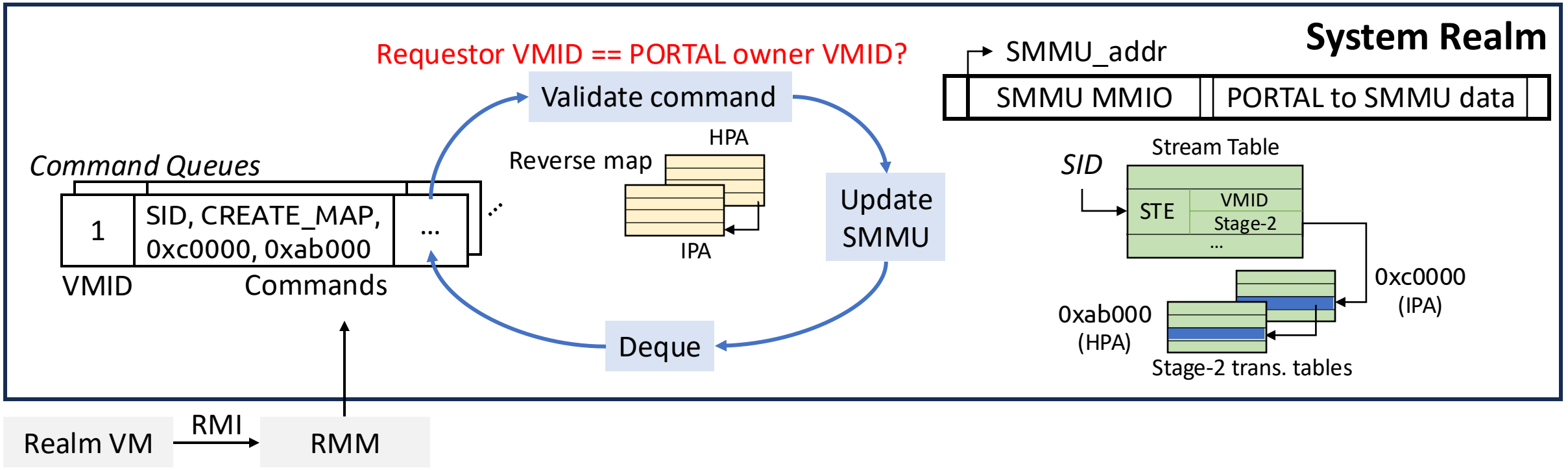


System Realm Command Queue



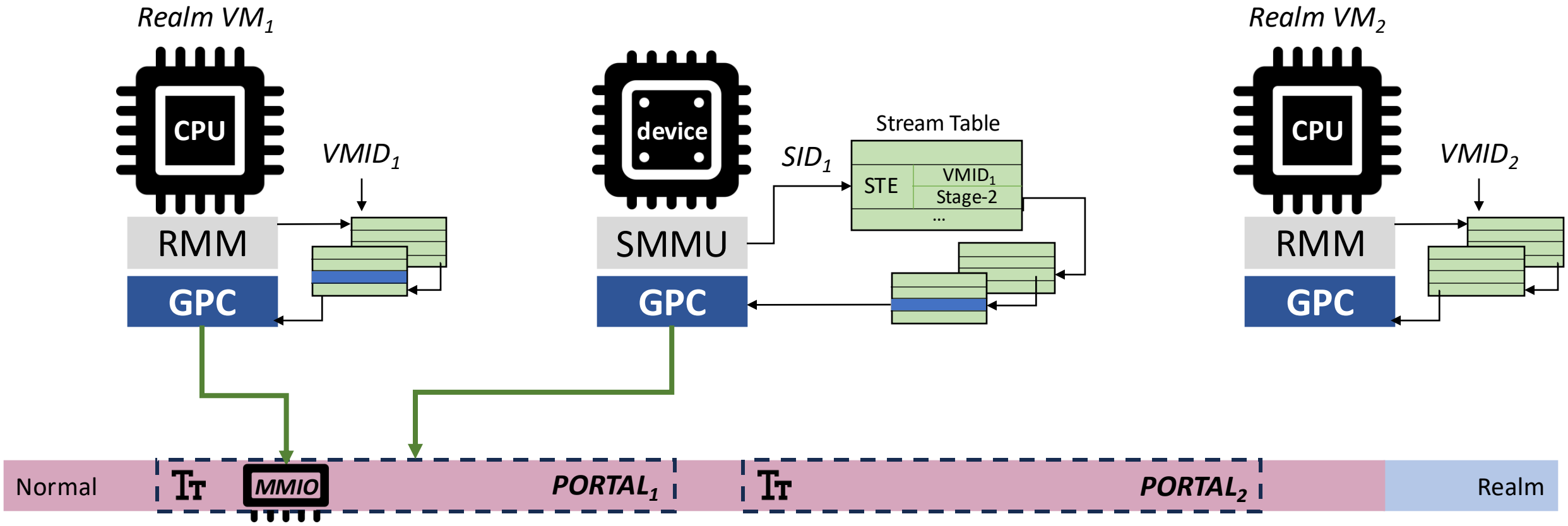
Execution Loop for Command Handling

Establish a Portal region between a Realm VM and a device

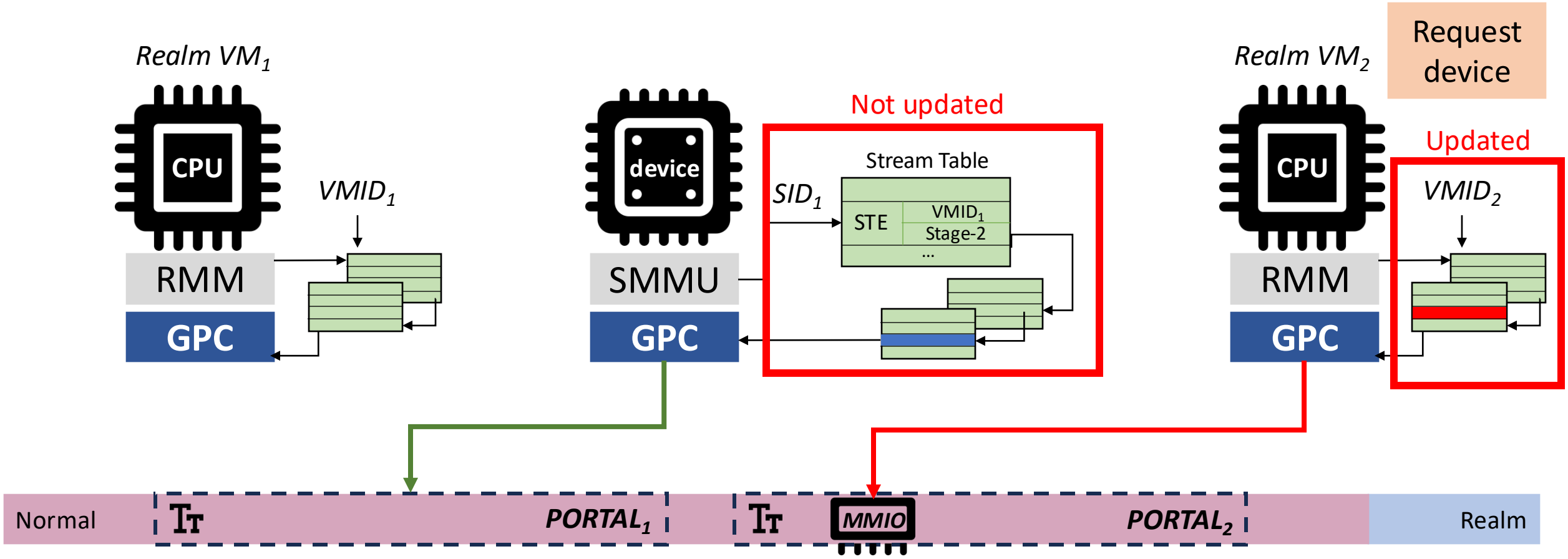


Device Management

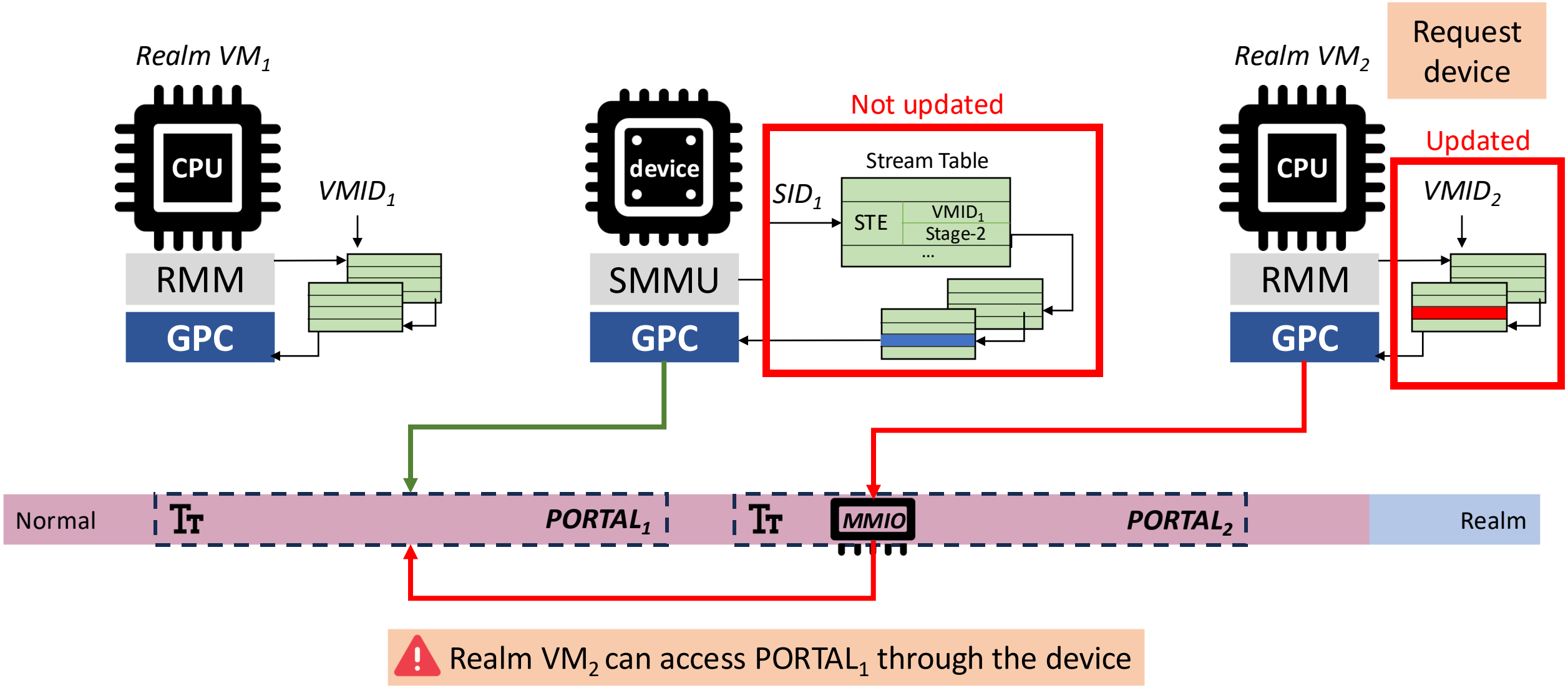
Careless Management



Careless Management

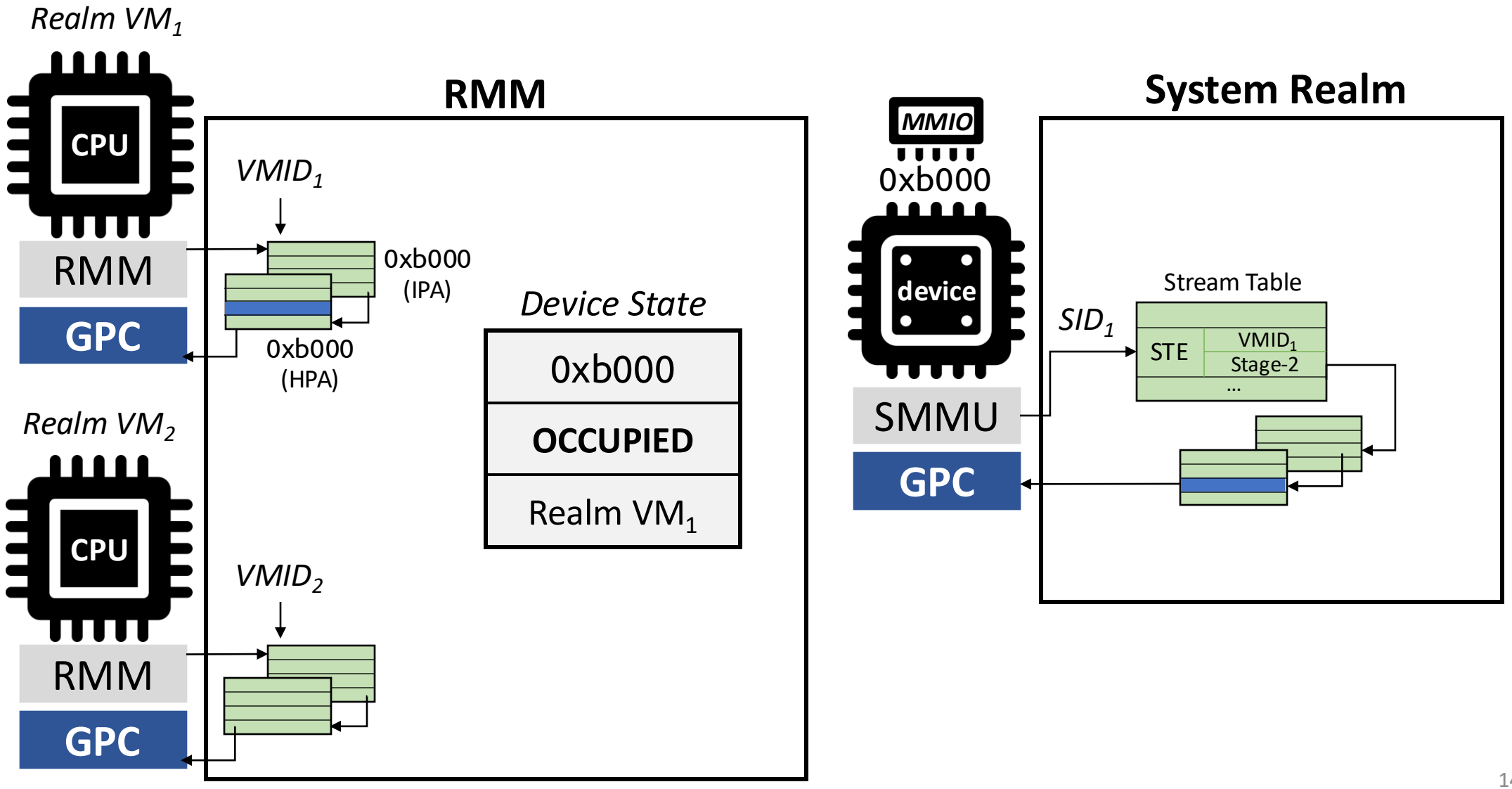


Careless Management

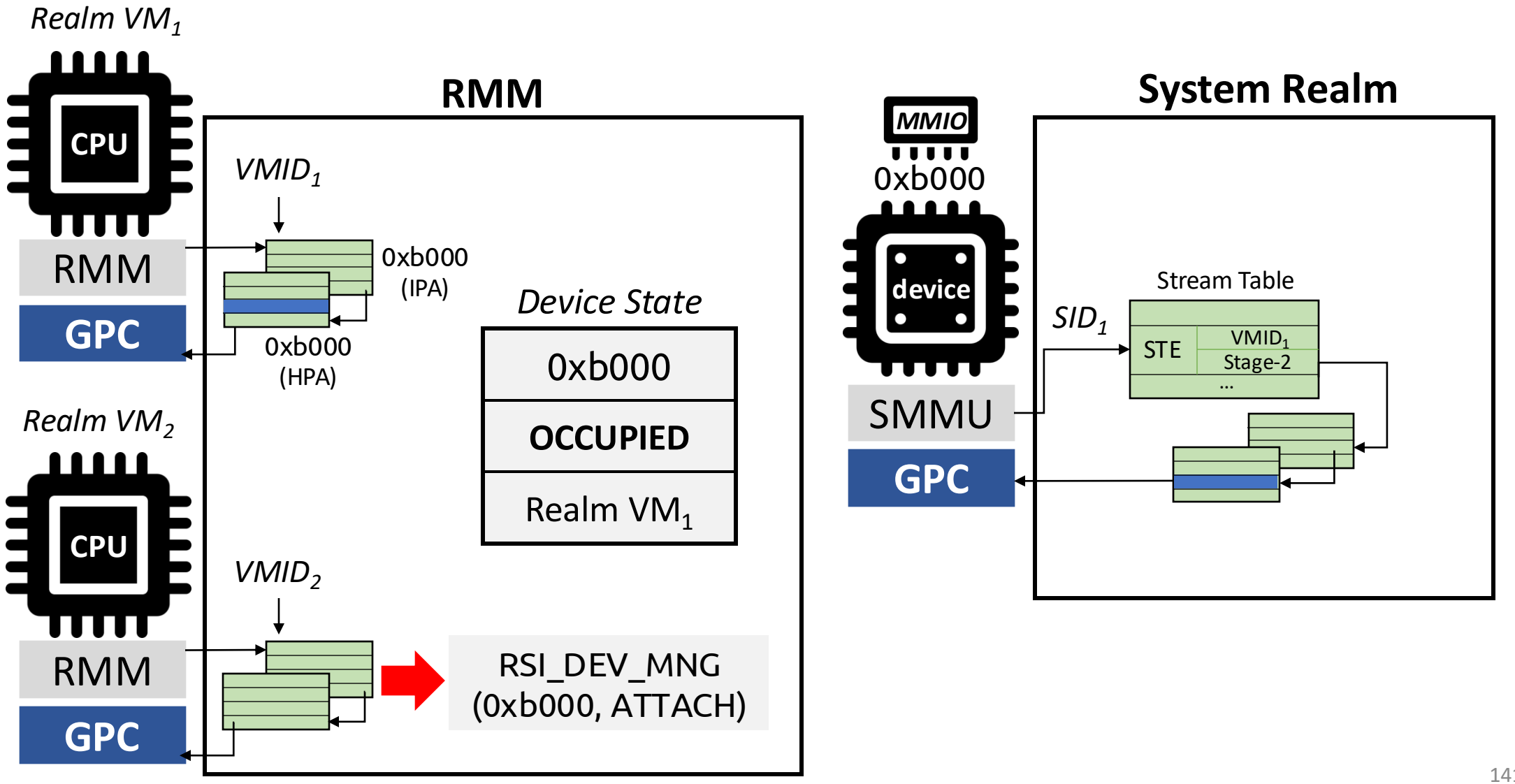


Realm VM_2 can access $PORTAL_1$ through the device

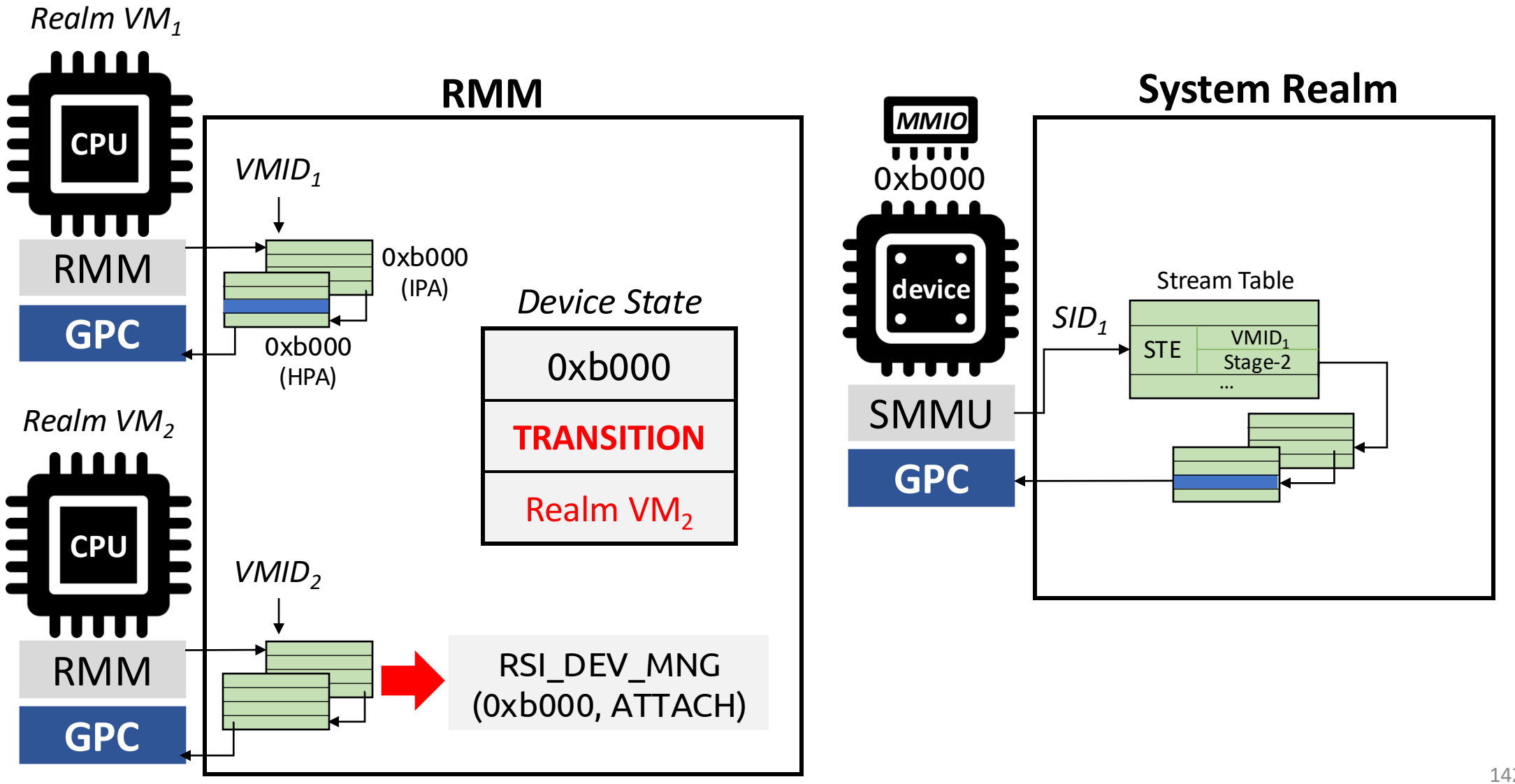
Portal Device Management



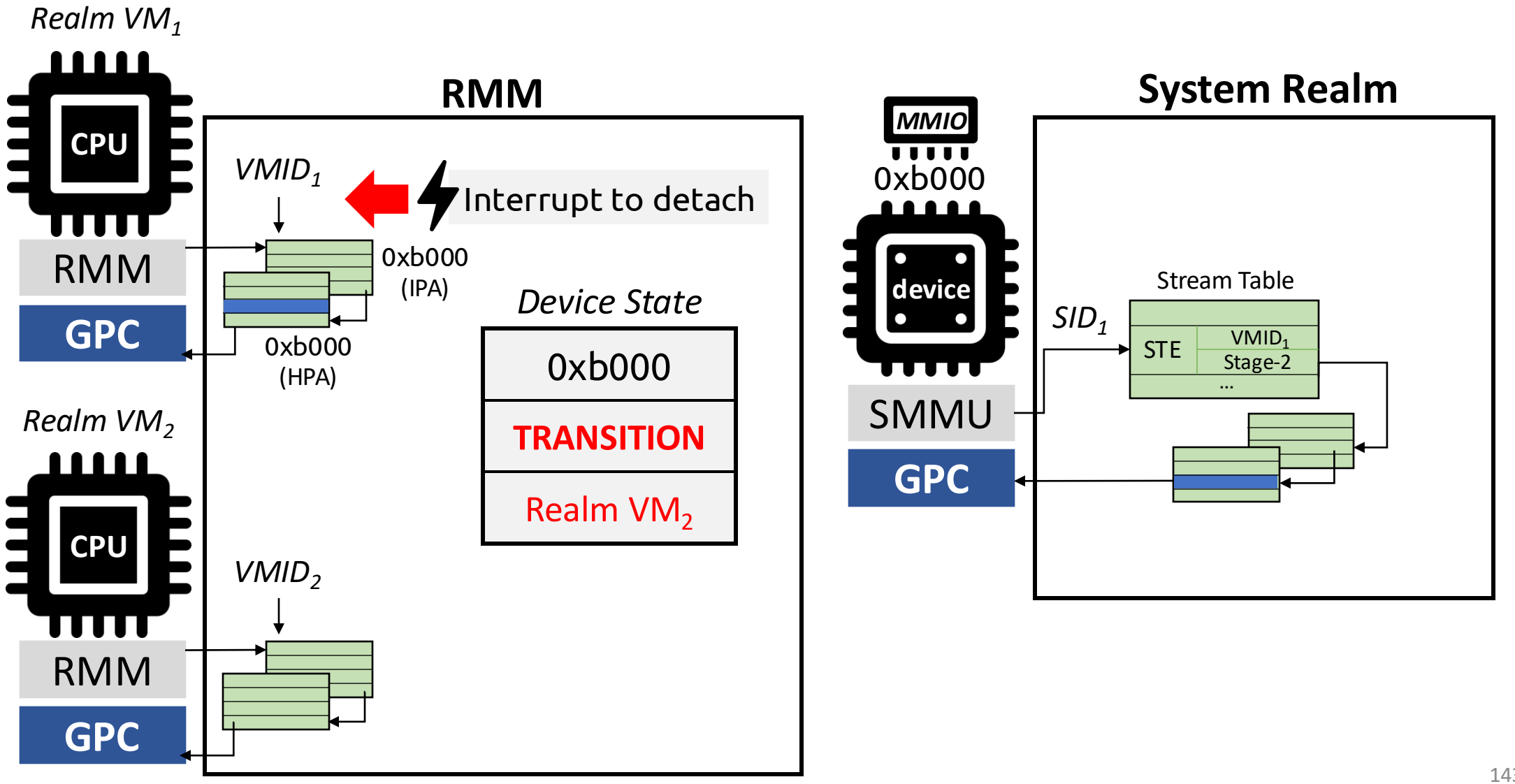
Portal Device Management



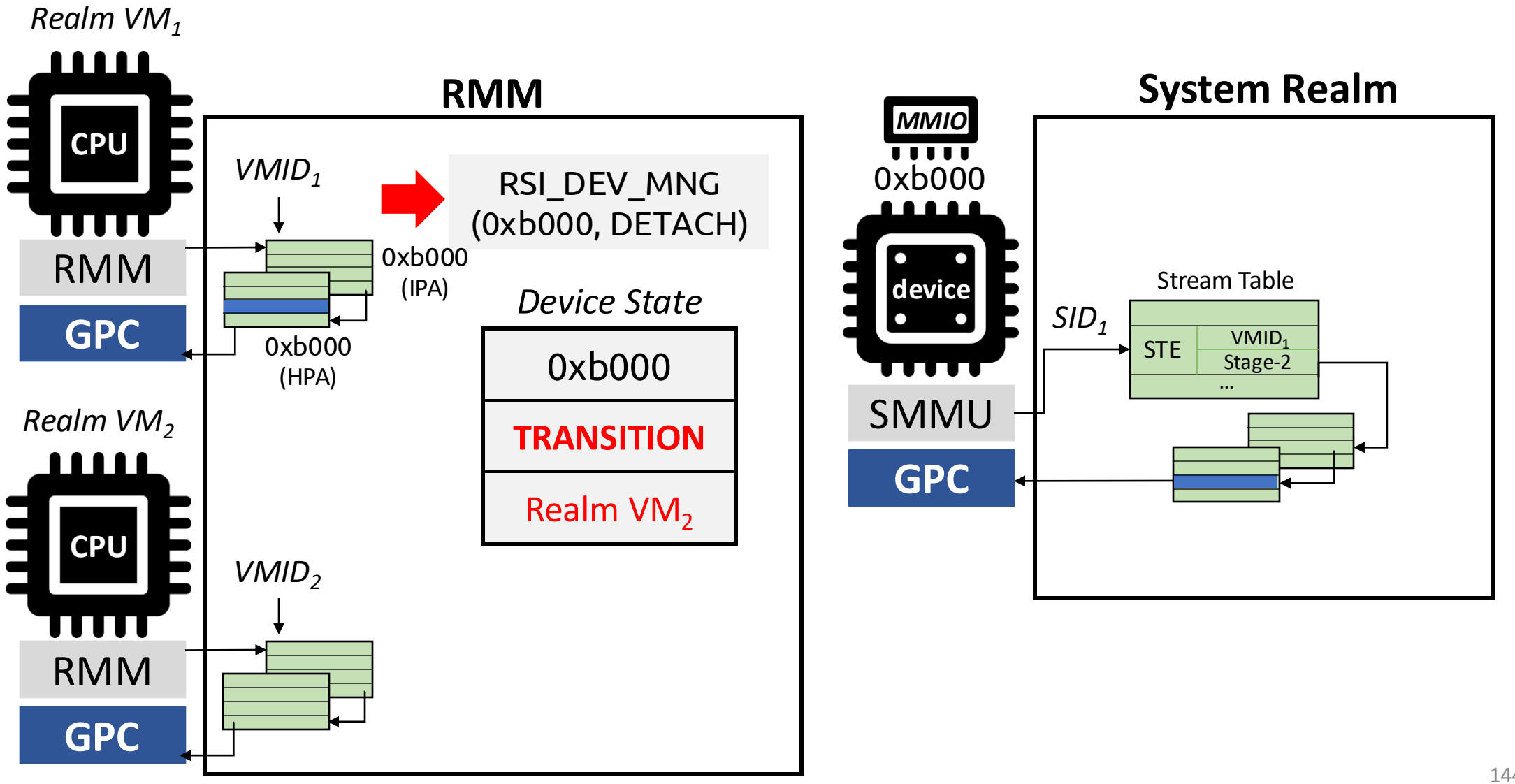
Portal Device Management



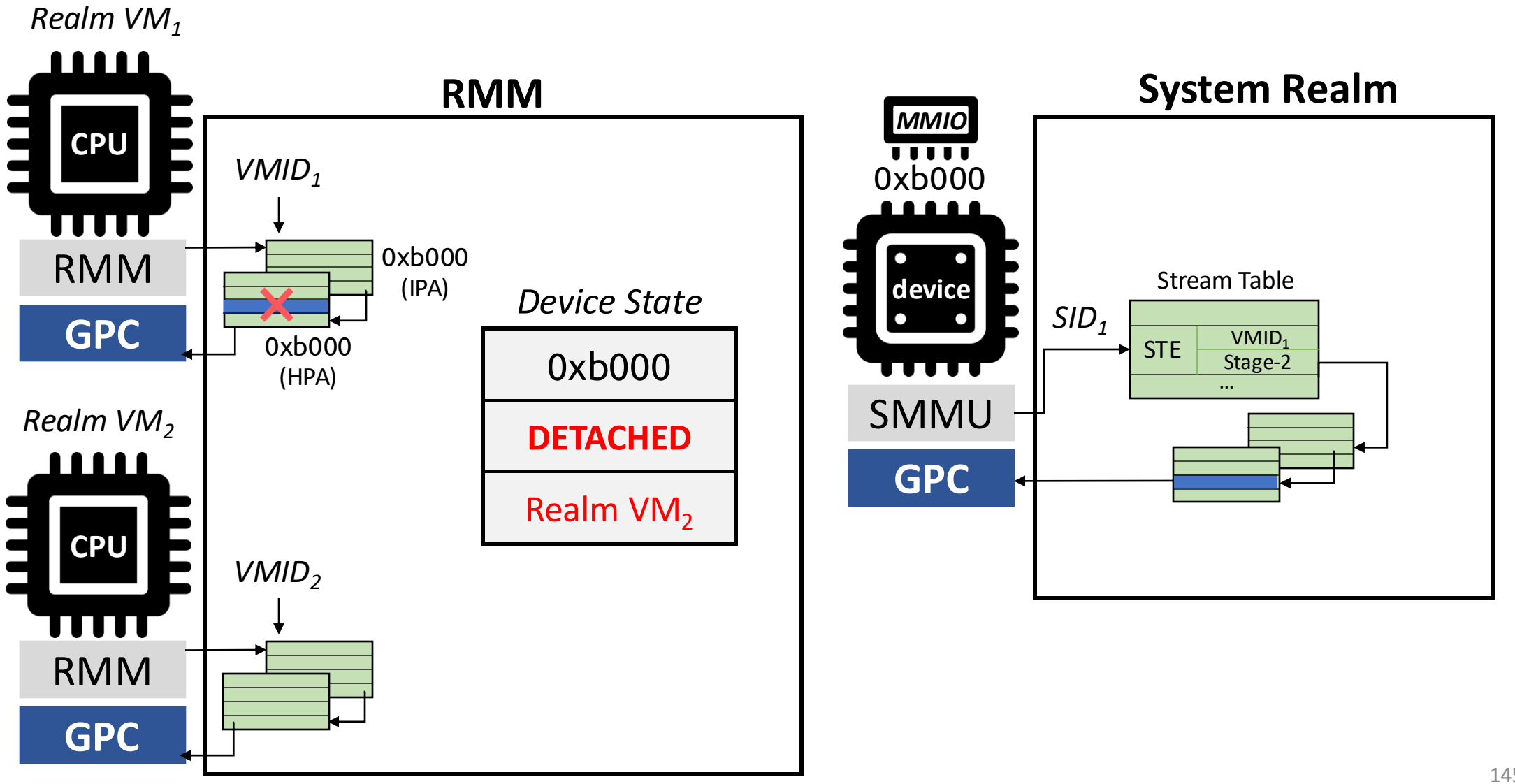
Portal Device Management



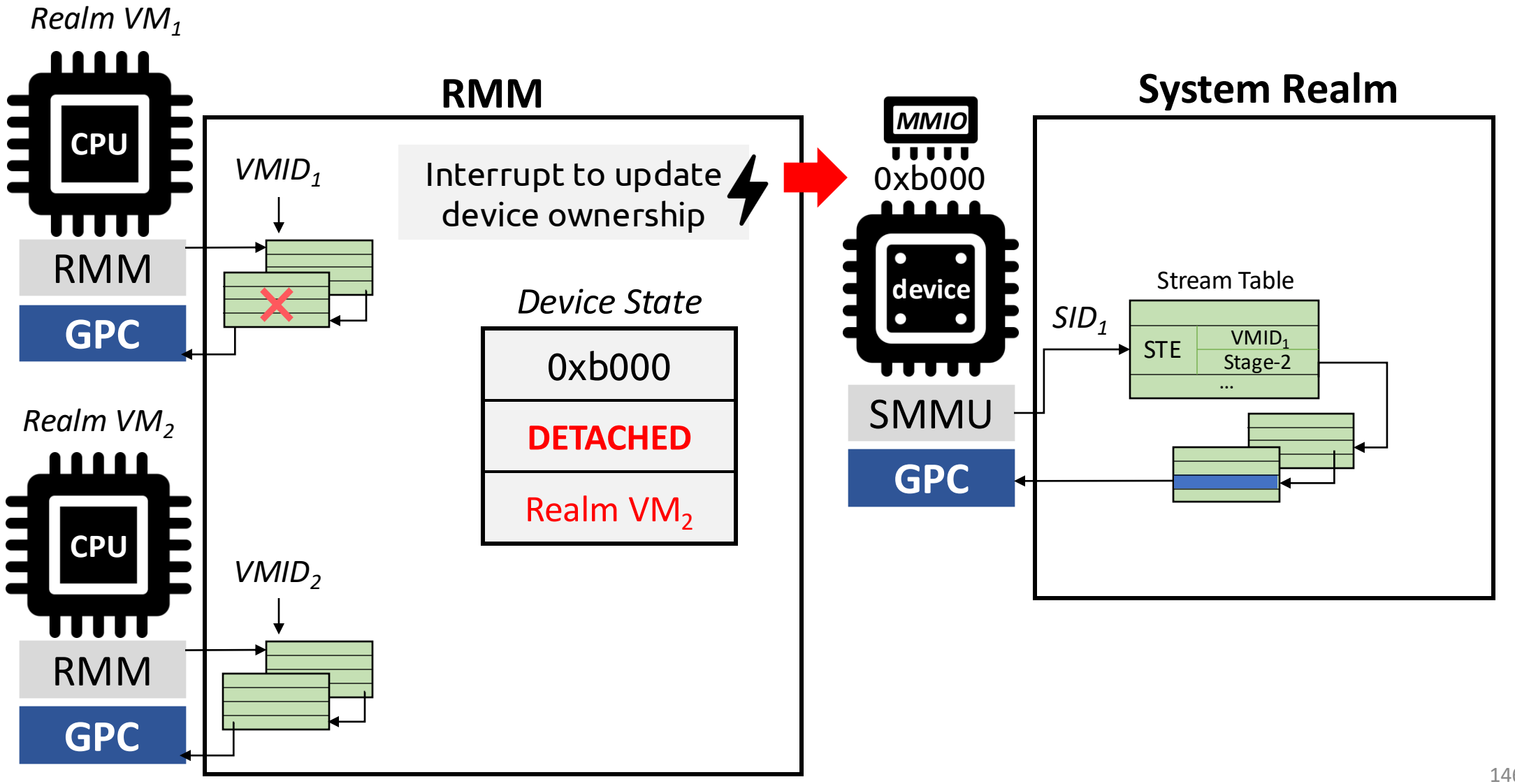
Portal Device Management



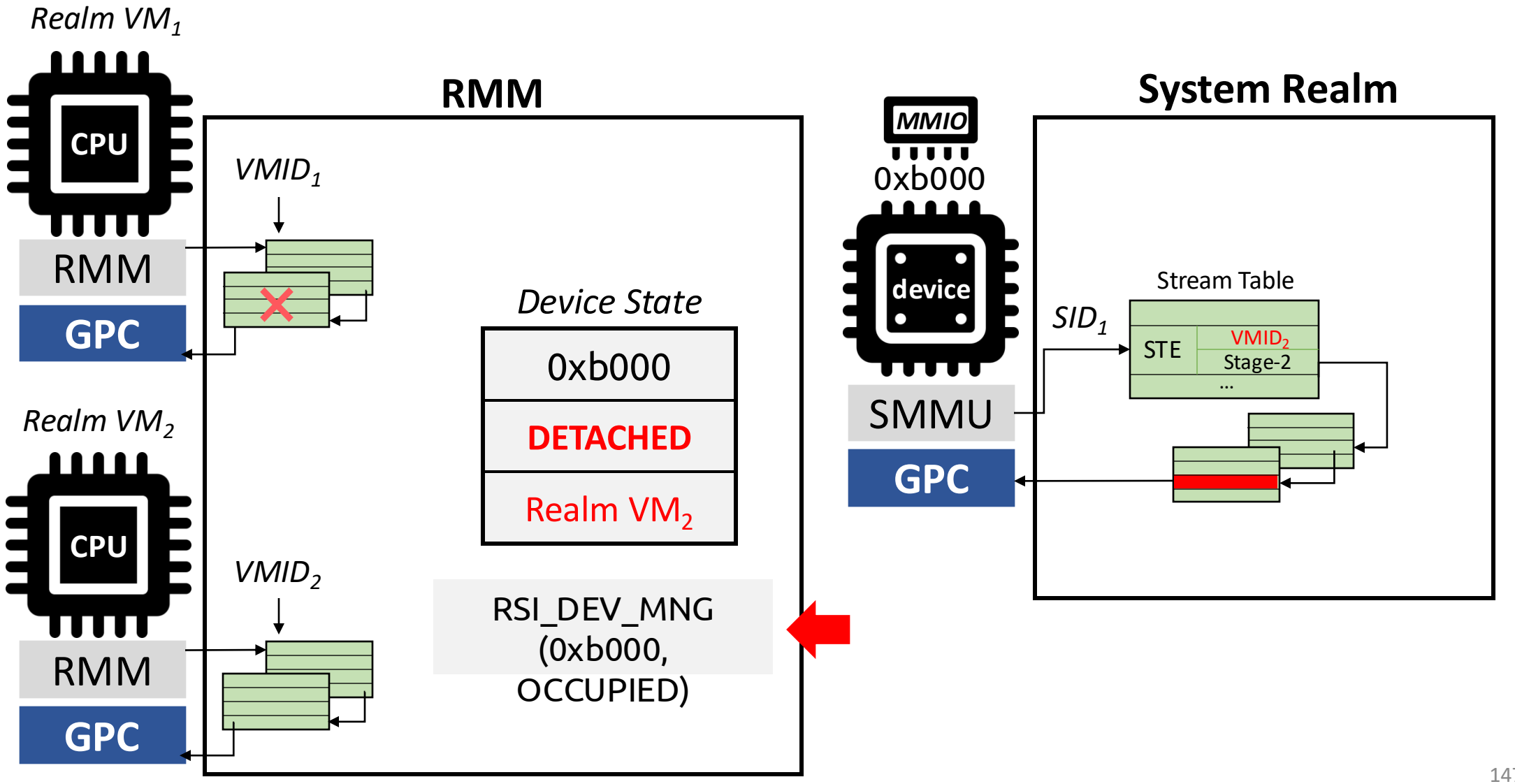
Portal Device Management



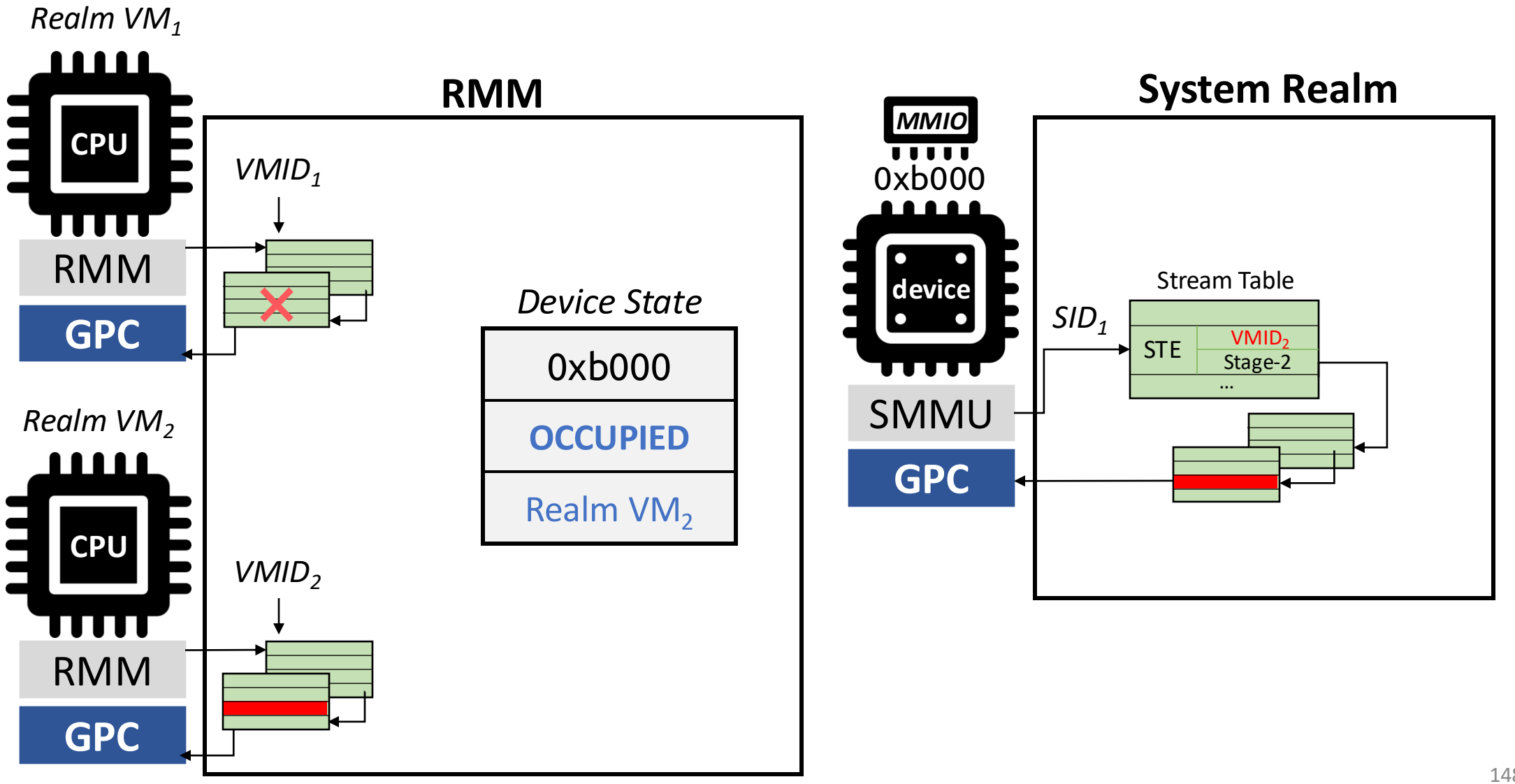
Portal Device Management



Portal Device Management



Portal Device Management



Implementation

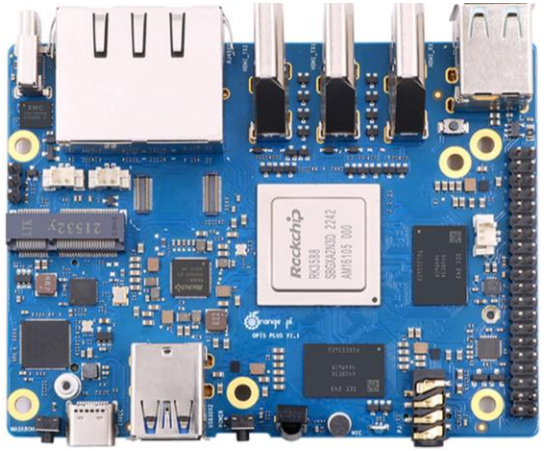
Functionality Prototype

- Arm FVP_Base_RevC2XAEMvA with RME support
- A connected test engine
  - Simulates a DMA-capable peripheral
  - An SMMU that supports RME
- **0.5MB** memory for device GPTs
- A reference implementation of the System Realm

Implementation

Performance Prototype

- Migrate FVP prototype to Orange Pi 5 Plus
  - RK3588 SoC
  - 8-core 64-bit Arm processor (4-core A76 and 4-core A55)
  - Arm Mali-G610 GPU
  - 8GB of shared DRAM
- Emulation of Arm CCA (Armv9) with Armv8 features



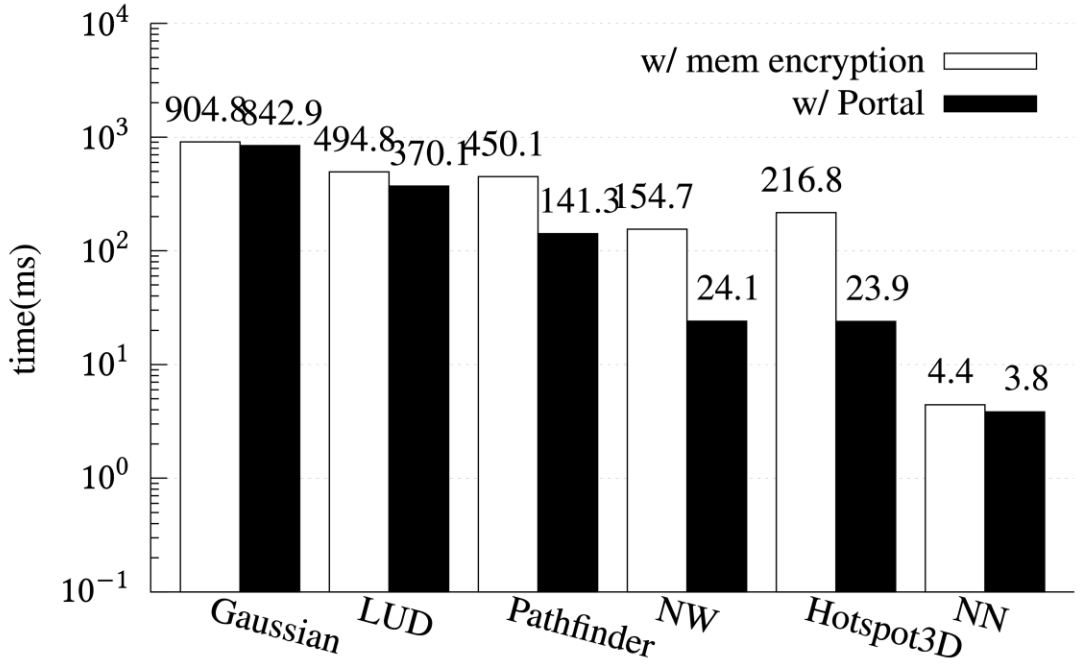
Performance Evaluation

- Performance of GPU tasks
- Rodinia GPU benchmark
- AES-GCM-based encrypted memory
- **3.71× (1.07×-9.07×)** performance

Overhead due to memory encryption
- Simpler processing logic
- Larger data size

Portal's advantage
+ Data intensive applications
+ Frequent transmission of larger data size

Application	Problem Size	Data Buffers	Memory
Gaussian	1024 × 1024 nodes	3	8.39 MB
LUD	2048 × 2048 nodes	1	16.00 MB
Pathfinder	100000 × 100 points	4	40.46 MB
NW	2048 × 10 nodes	2	16.79 MB
Hotspot3D	512 × 512 × 8 nodes	3	25.16 MB
NN	42764 nodes	2	0.51MB



Performance of Portal Lifecycle

- Initializing GPTs
  - One-time overhead for P-GPTs
- Initializing Realm VM
  - No overhead if no Portal support
  - **2%** overhead of creating the command queue
- Entering Realm VM (i.e., RMI_REC_ENTER)
  - **6.3%** overhead if device management is needed

Performance of System Realm

- Initializing System Realm
  - Instantiated during host OS boot
  - Persists until system shutdown
  - **One-time 1.5%** overhead
- Processing SMMU management commands
  - **2 µs** of world switching to enter System Realm
  - **1.9 µs** to **2.4 µs** to process 20 commands for two Realm VMs

Performance of Device Management

- Device initial attachment
  - **46.29 µs** to attach Mali-G610 GPU (512 MMIO pages)
- Device reassignment
  - **175 µs** overhead due to inter-Realm VMs communication
  - Costly but not frequent
  - Current RMM relies on VMM for MMIO access

Memory Overhead

- Maintaining P-GPT (for a 4GB memory region)
  - Two-level table
    - 1GB for L0 regions and 4KB for granule size
  - A **4KB** page in SRAM (L0 GPT)
  - **0.5MB** of DRAM (L1 GPT)

Memory Overhead

- Device management
  - **8 bytes** for each managed device state
- Command queue
  - A **256KB** command queue per Realm VM
    - 16-byte entry size, up to 16,384 requests

Portal Summary

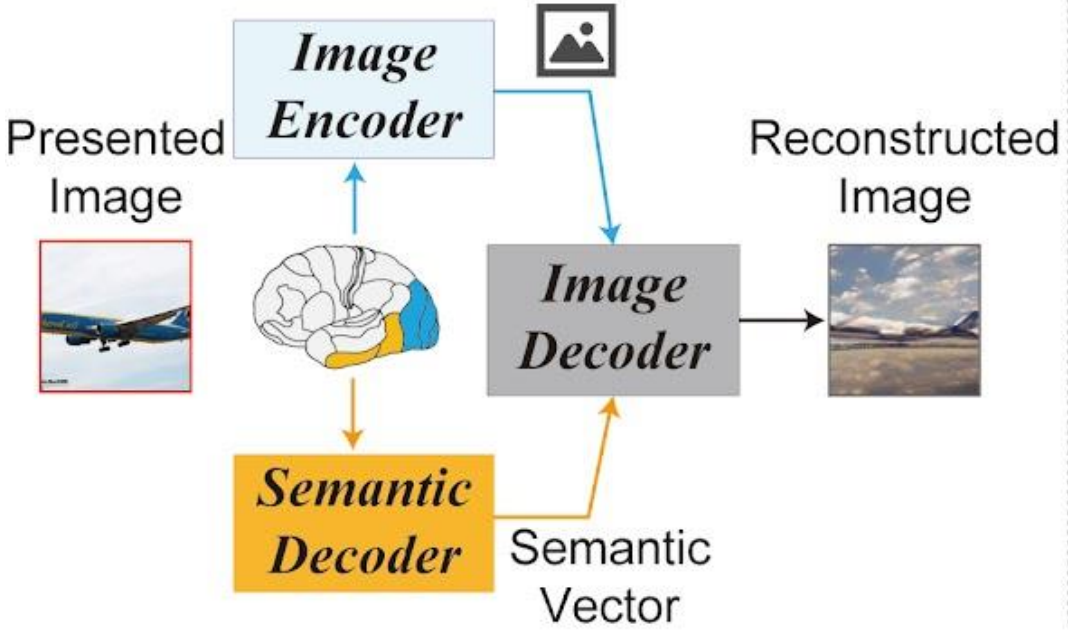
- Arm SoCs undergo a trend of increasing number of integrated devices
- Memory encryption hinders wider adoption of Arm CCA on mobile SoCs
- The integrity guarantee of SoCs vs. memory encryption
- Portal achieves secure device I/O without memory encryption
  - GPC and SMMU for hardware-level memory access control
  - Dynamic device management via System Realm
  - Better performance, scalability, and power efficiency for mobile SoCs
- Efforts on meeting QoE requirements for VR + TEE

Discussion and Future Work

- Remove memory encryption for Realm VMs on Arm SoC
  - A top-down approach
- Discrete devices
  - PCIe-5 link-layer encryption
- Import a real VR system onto Arm CCA with Portal
  - Arm CCA hardware available end of year
  - Adapt open-source comprehensive VR system simulation

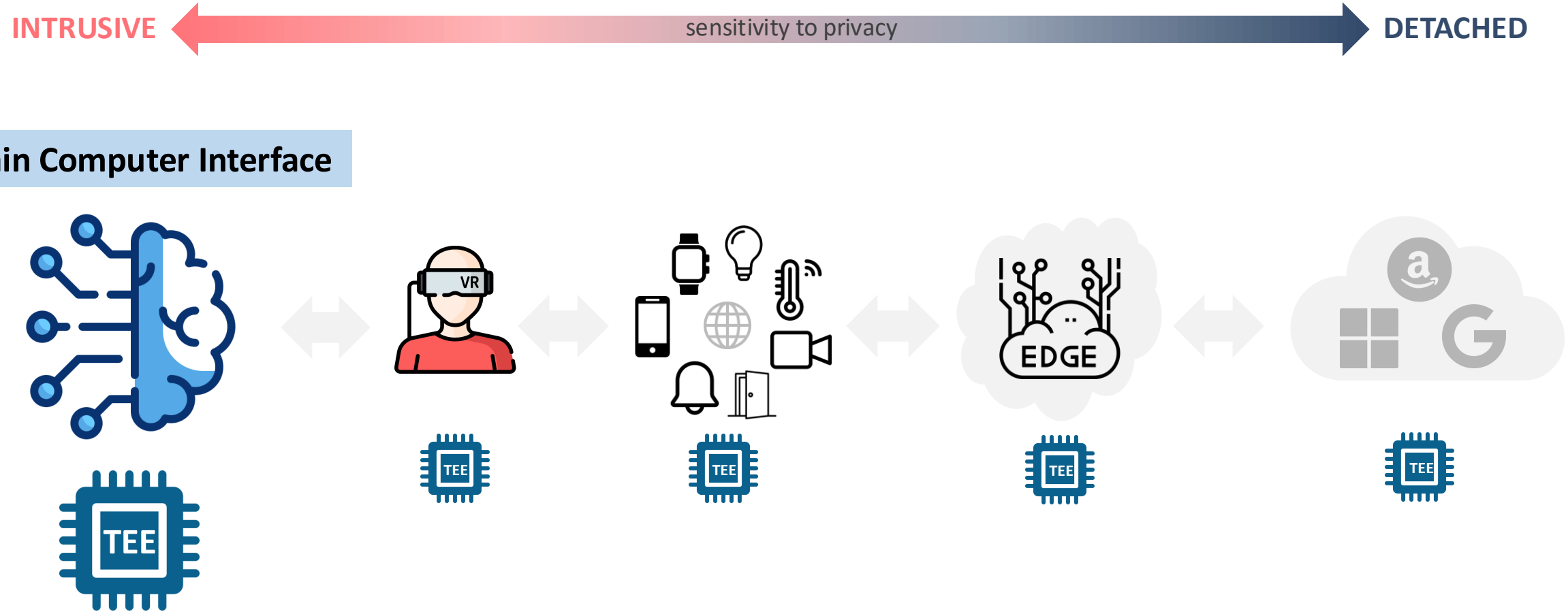
Future Research: Motivation

High-resolution image reconstruction with latent diffusion models from human brain activity [1]

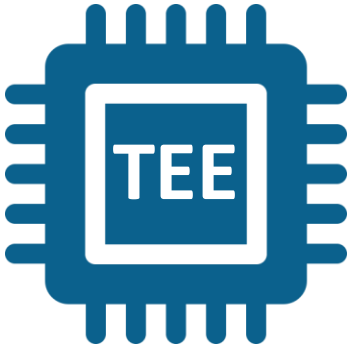


[1] Takagi, Yu and Nishimoto, Shinji, "High-resolution image reconstruction with latent diffusion models from human brain activity," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, Canada, Jun. 2023.

Future Research: TEE for BCI (Silicon-Carbon)



Conclusion



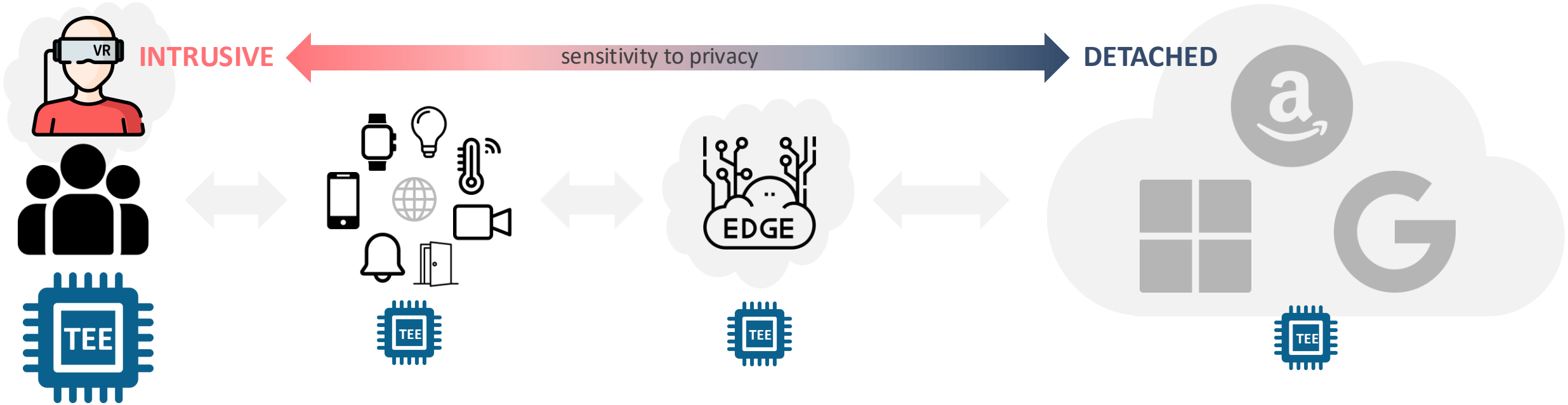
Harden Current TEEs

Understand Security of VR

Adapt with Requirements

Conclusion

Adapt (Secure) TEEs for (Insecure) VR Platforms



Q&A

Thank you!

Ph.D. Defense of Dissertation

Fan Sang

Advisor: Prof. Taesoo Kim

Georgia Tech College of Computing
School of Cybersecurity
and Privacy