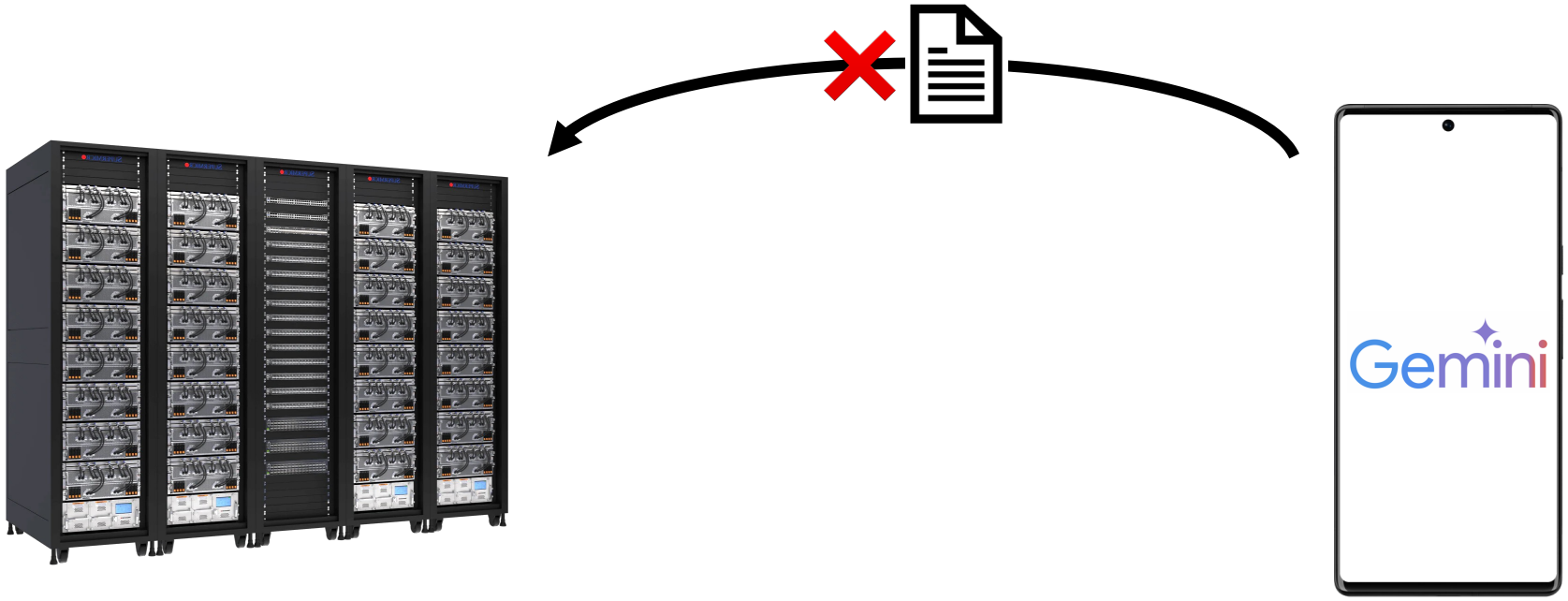
ASGARD: Protecting On-Device Deep Neural Networks with Virtualization-Based Trusted Execution Environments

Myungsuk Moon, Minhee Kim, Joonkyo Jung, Dokyung Song

YONSEI UNIVERSITY

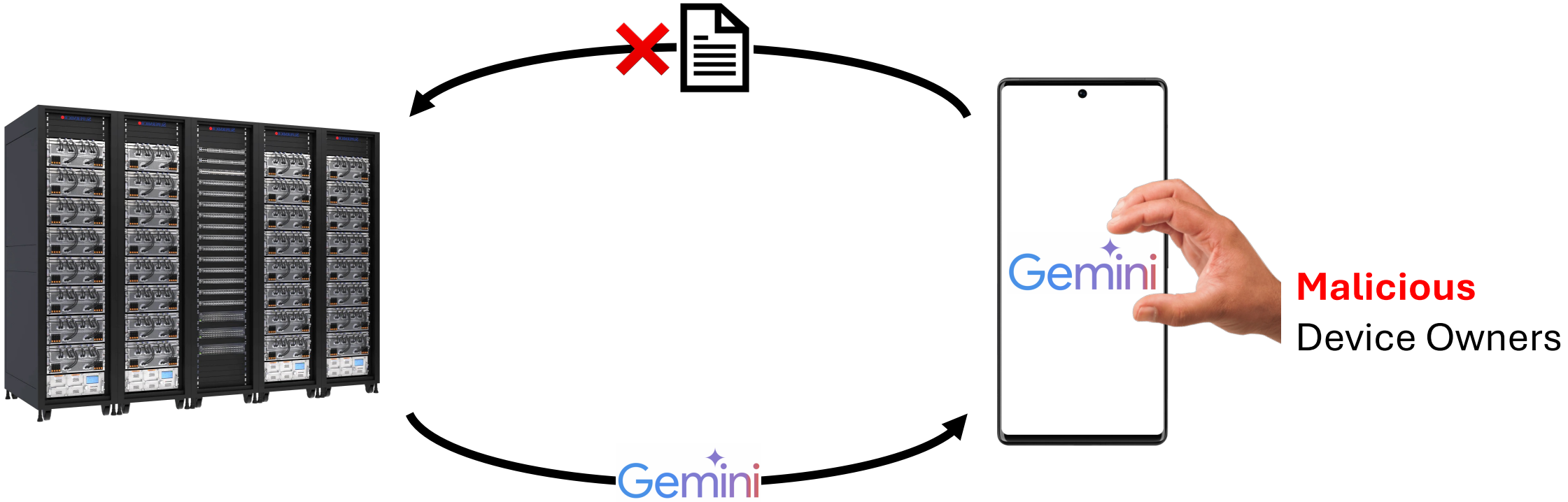
Problem: On-Device DNN Protection from Device Owners

1 **On-device AI protects user data**
by not sending it



Problem: On-Device DNN Protection from Device Owners

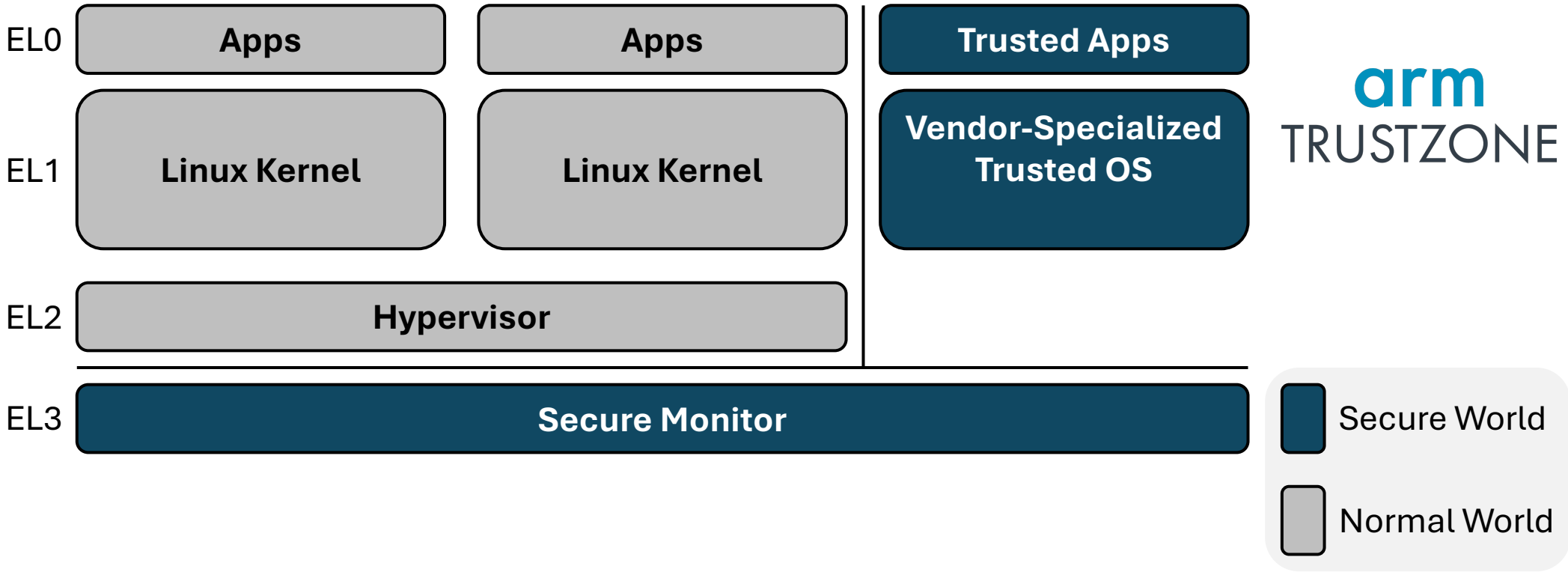
1 **On-device AI protects user data**
by not sending it



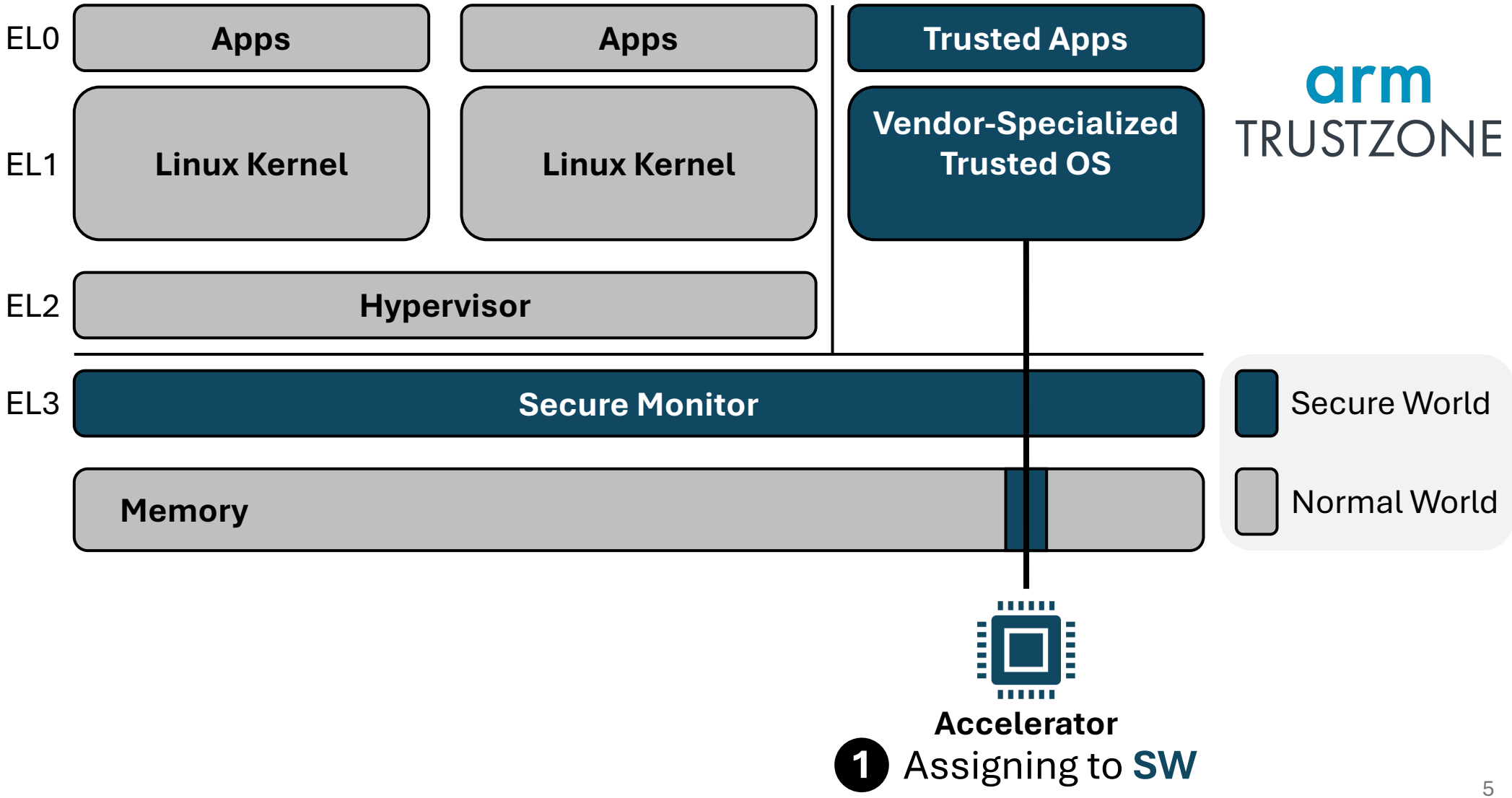
Malicious
Device Owners

2 **But on-device DNNs must be protected**
because they are sent to user devices

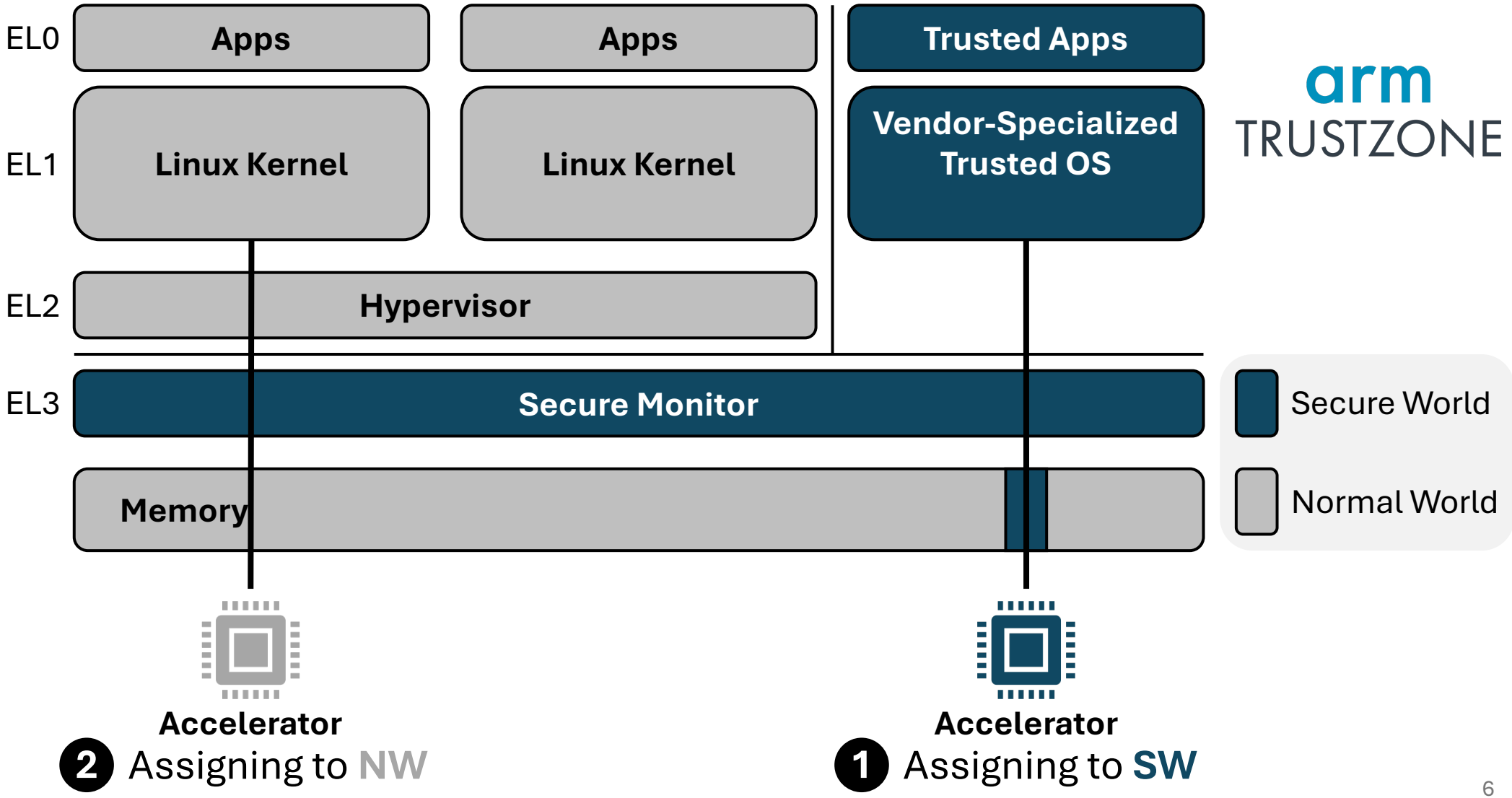
Prior Approaches with Arm TrustZone



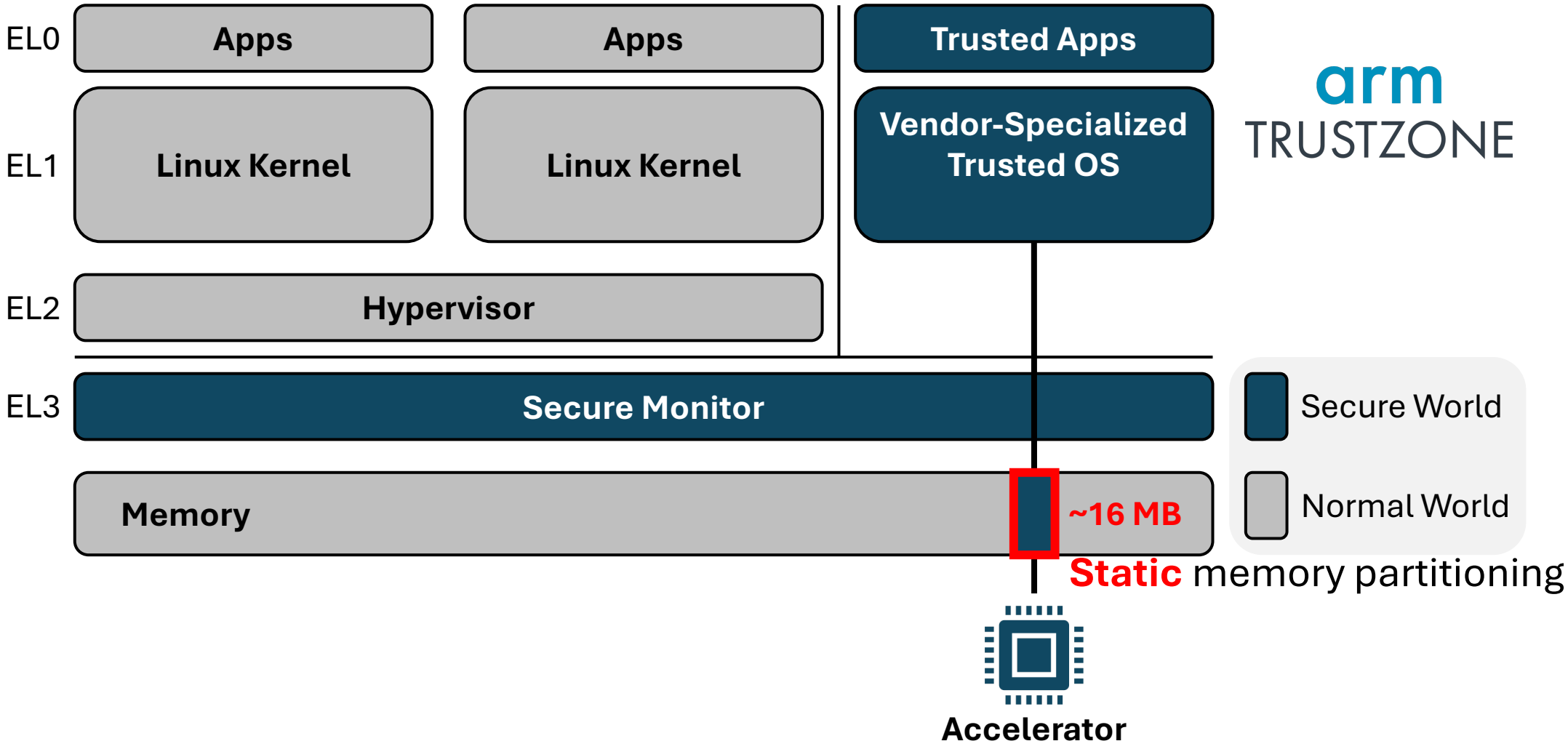
Prior Approaches with Arm TrustZone



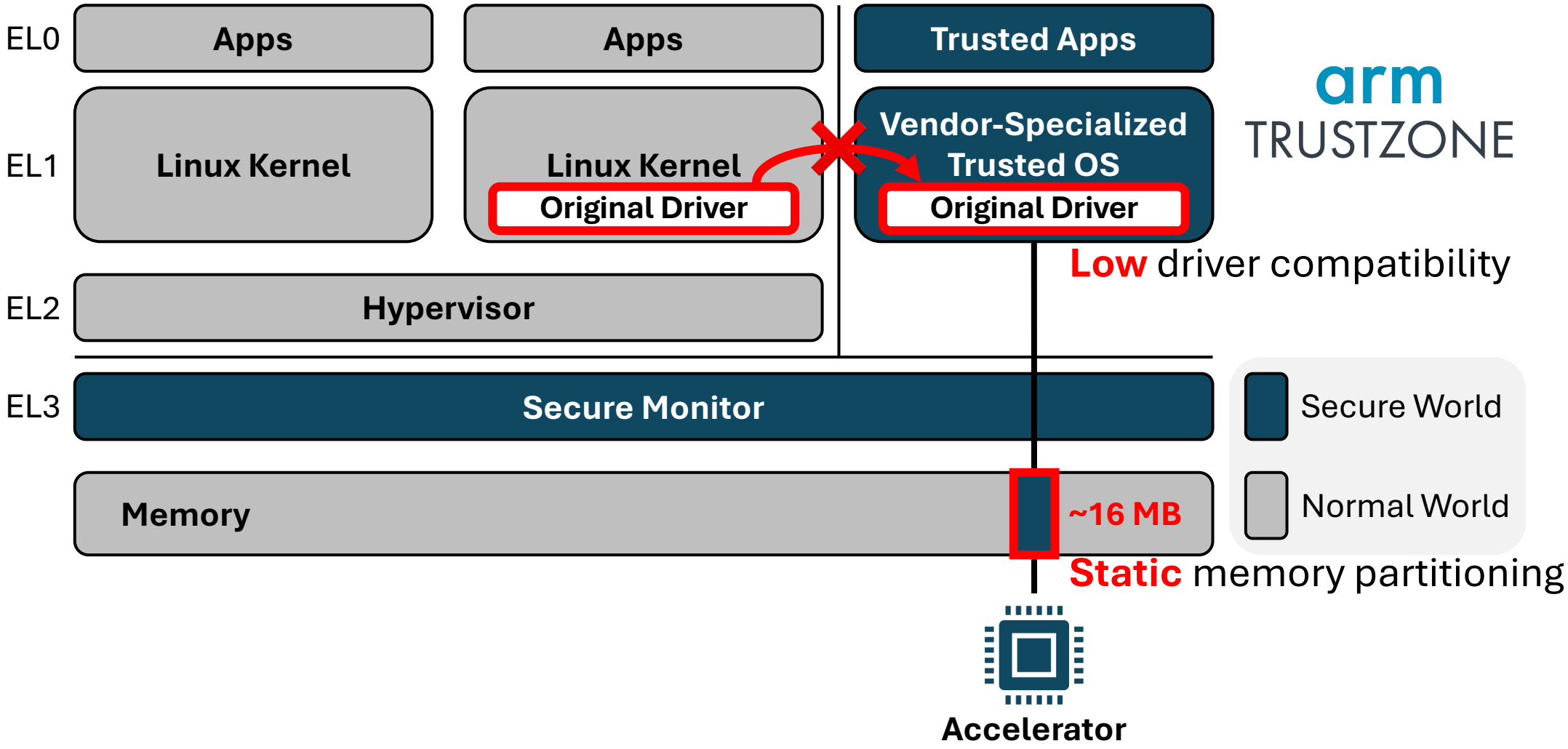
Prior Approaches with Arm TrustZone



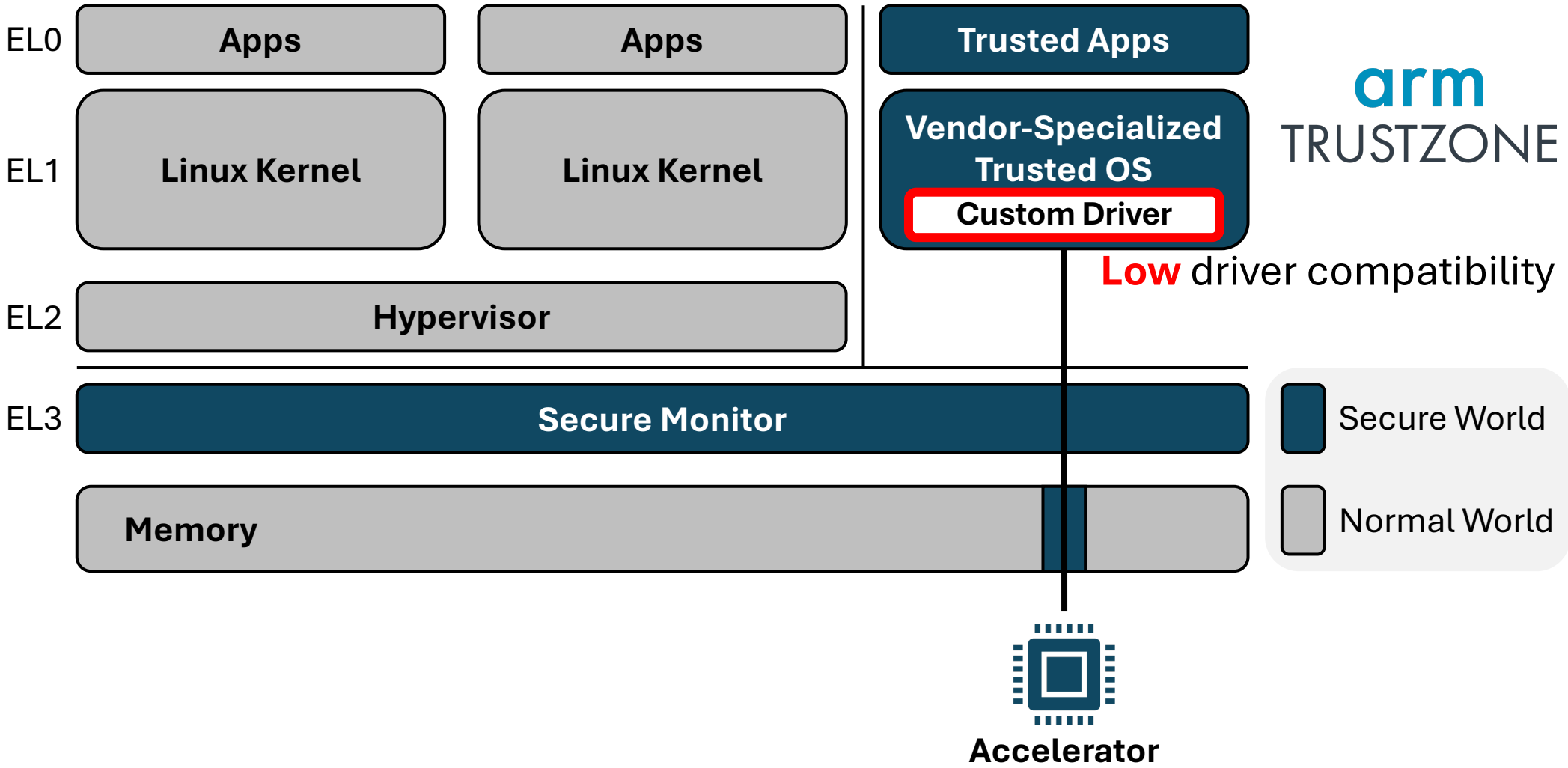
Prior Approach #1: Assigning Accelerator to Secure World



Prior Approach #1: Assigning Accelerator to Secure World

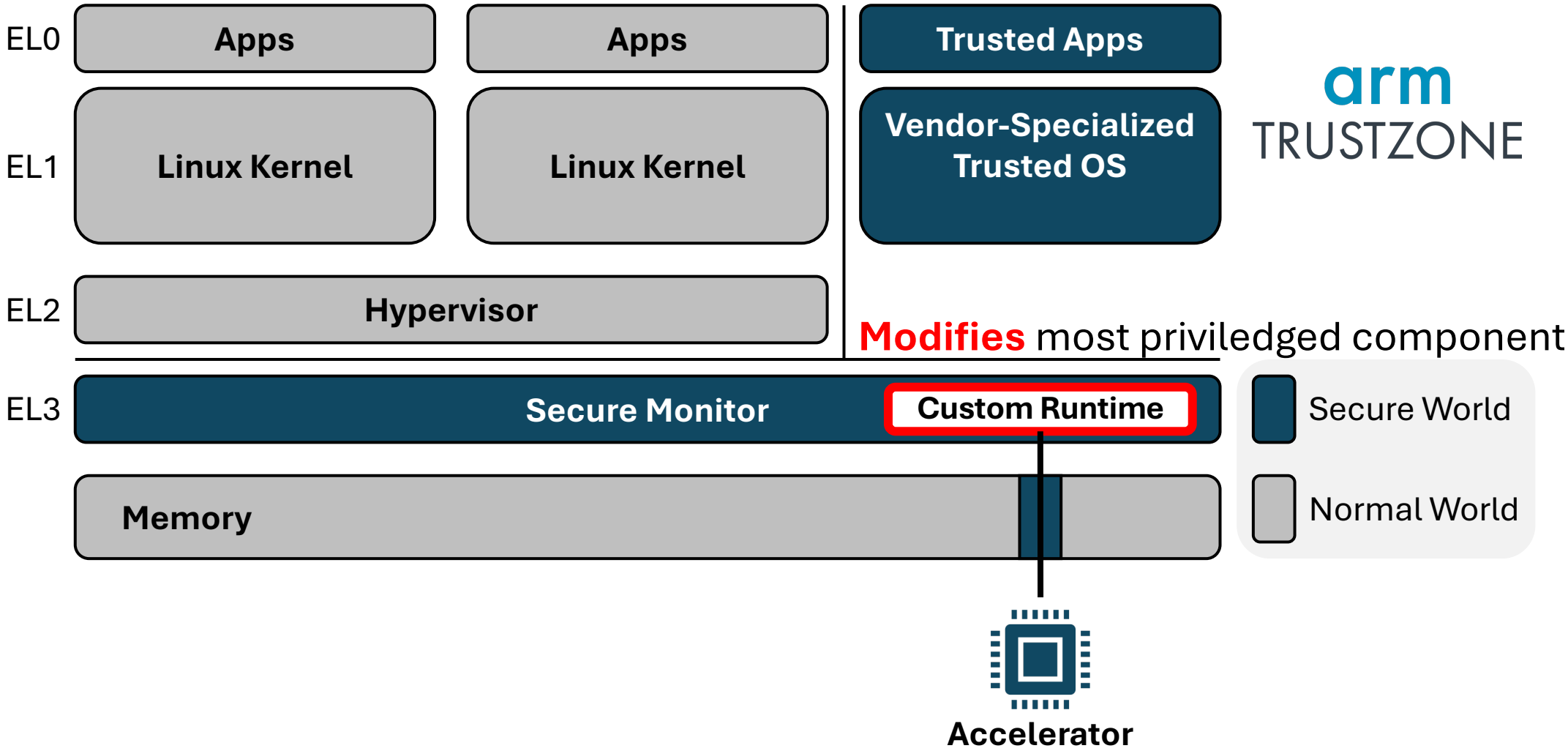


Prior Approach #1: Assigning Accelerator to Secure World



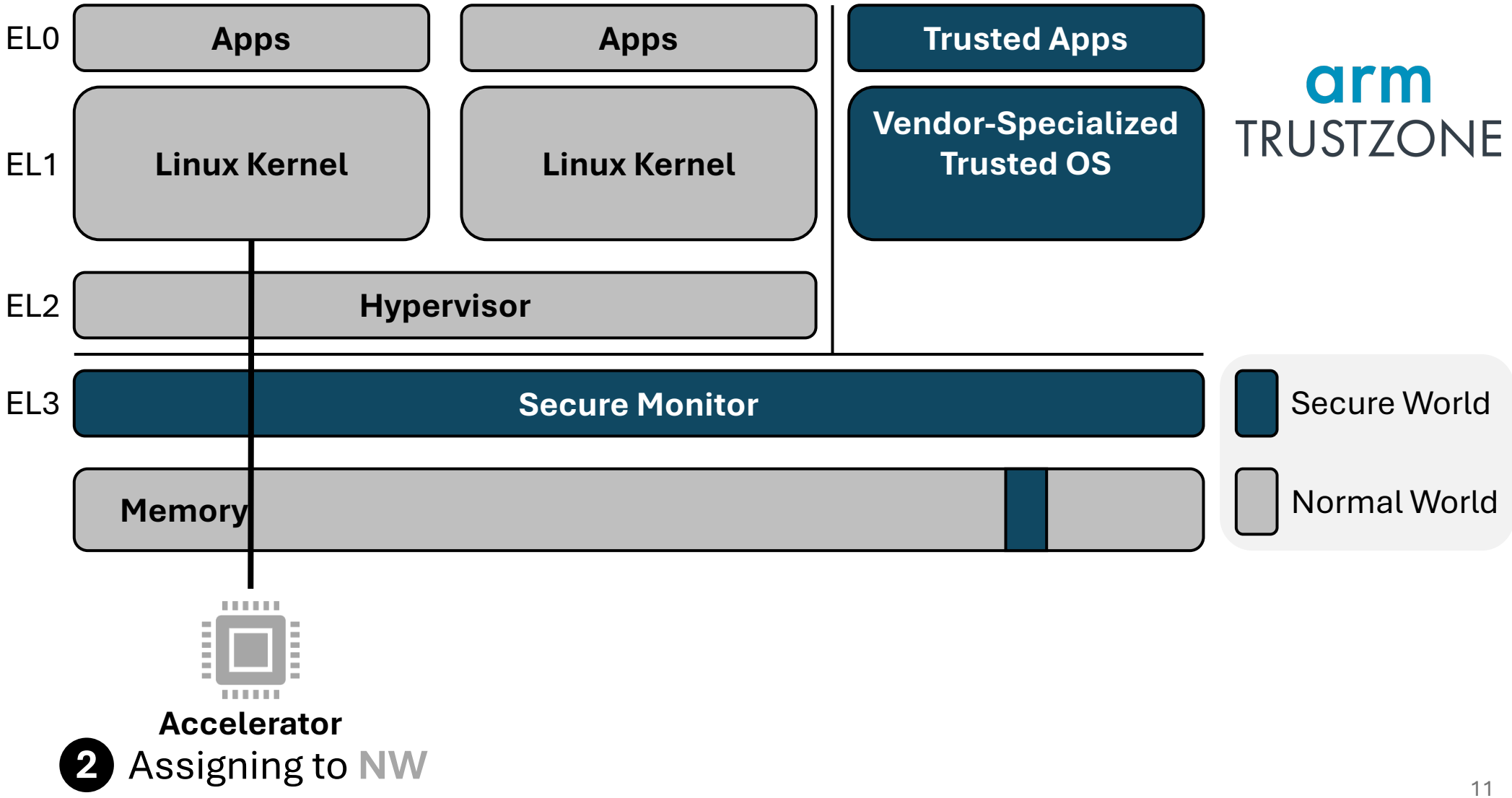
[1] Park and Lin, "GPUReplay: A 50-kb GPU stack for client ML," *ASPLOS '22*
[2] Guo and Lin, "Minimum viable device drivers for ARM TrustZone," *EuroSys '22*

Prior Approach #1: Assigning Accelerator to Secure World

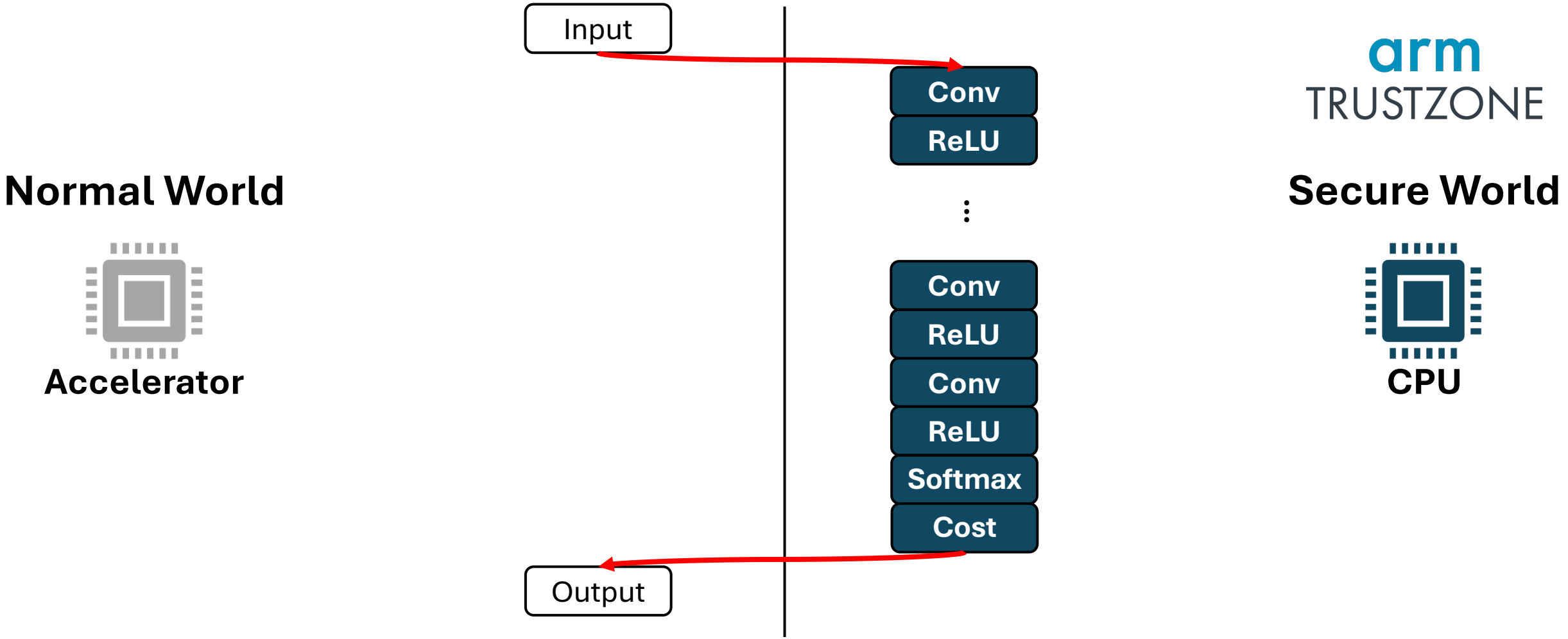


[3] Deng et al., "StrongBox: A GPU TEE on Arm endpoints," *CCS '22*

Prior Approach #2: Assigning Accelerator to Normal World

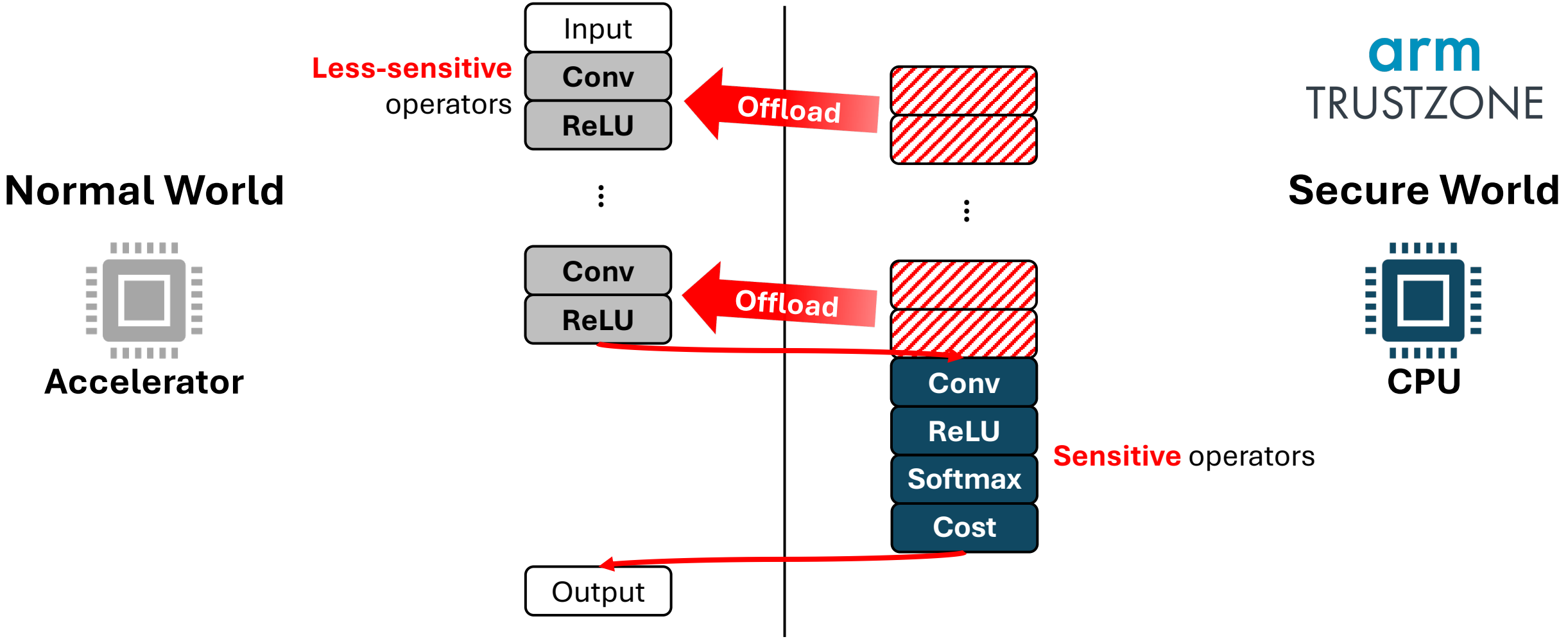


Prior Approach #2: Assigning Accelerator to Normal World



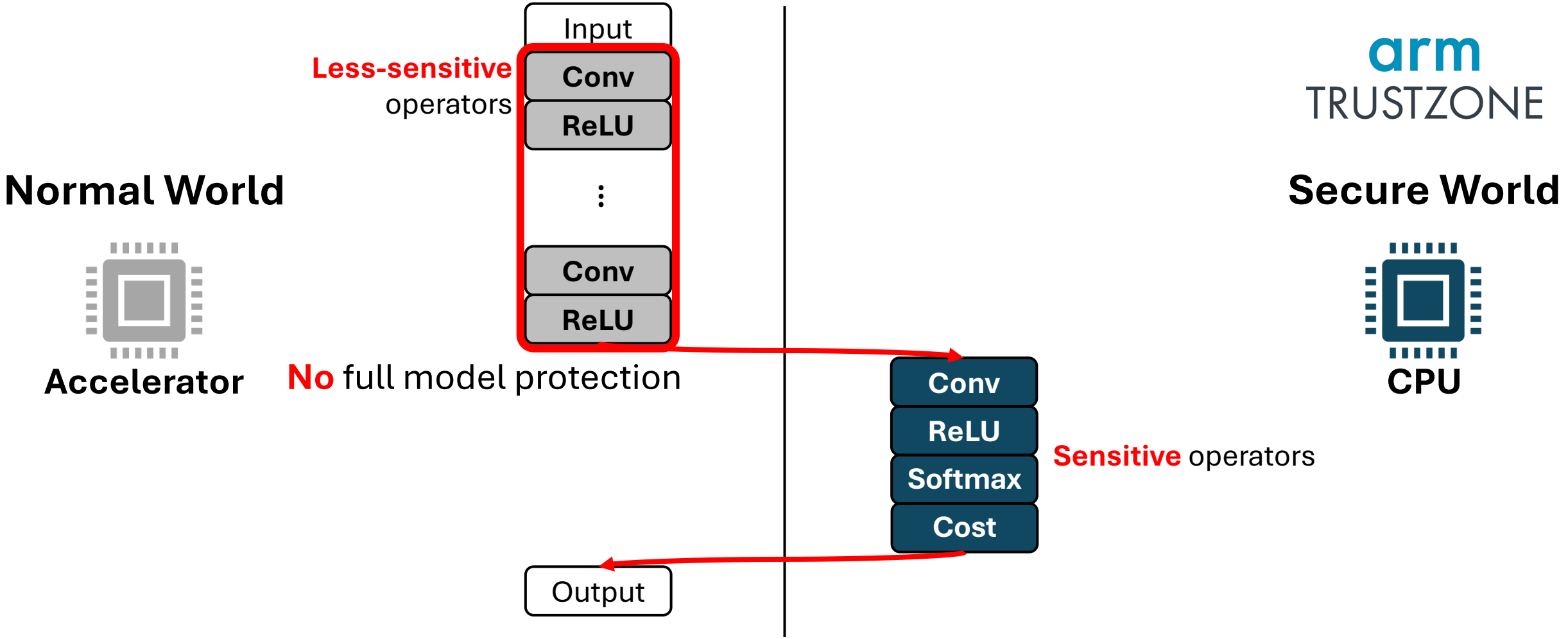
[4] Mo et al., "DarkneTZ: Towards model privacy at the edge using trusted execution environments," *MobiSys '20*

Prior Approach #2: Assigning Accelerator to Normal World



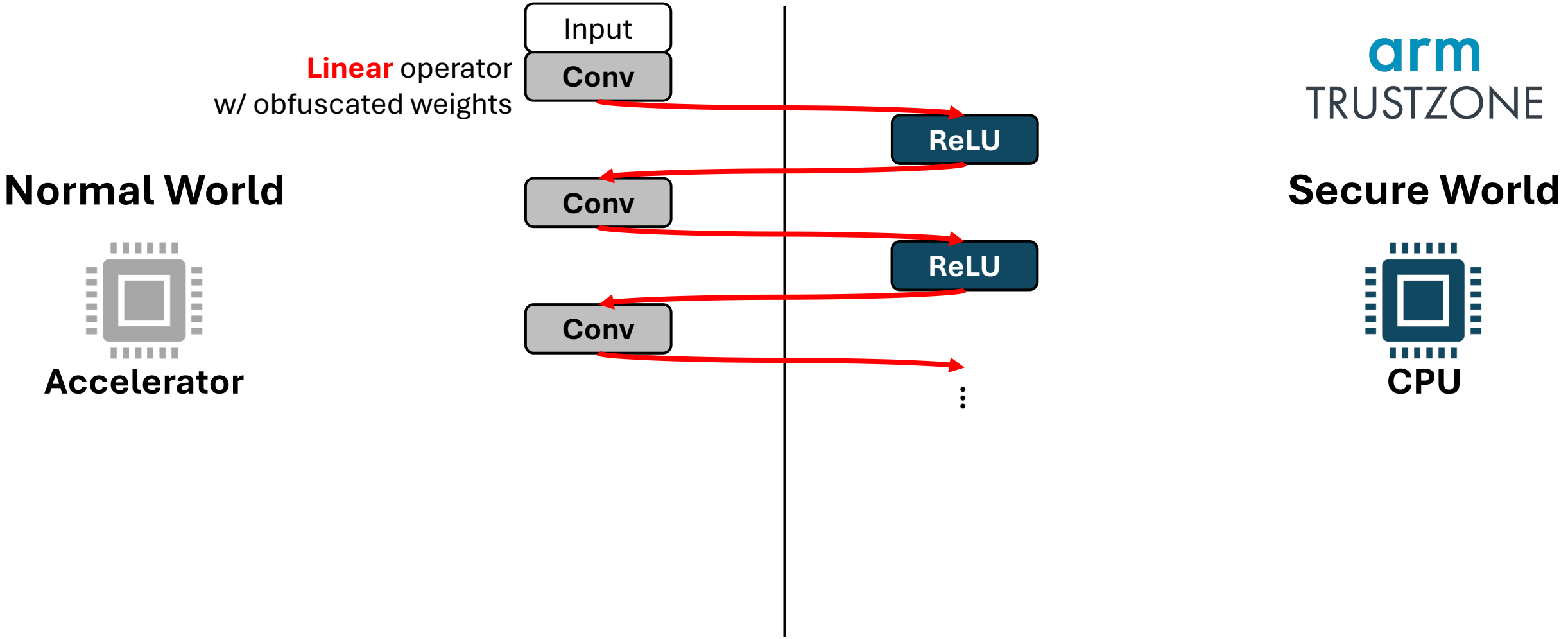
[4] Mo et al., "DarkneTZ: Towards model privacy at the edge using trusted execution environments," *MobiSys '20*

Prior Approach #2: Assigning Accelerator to Normal World



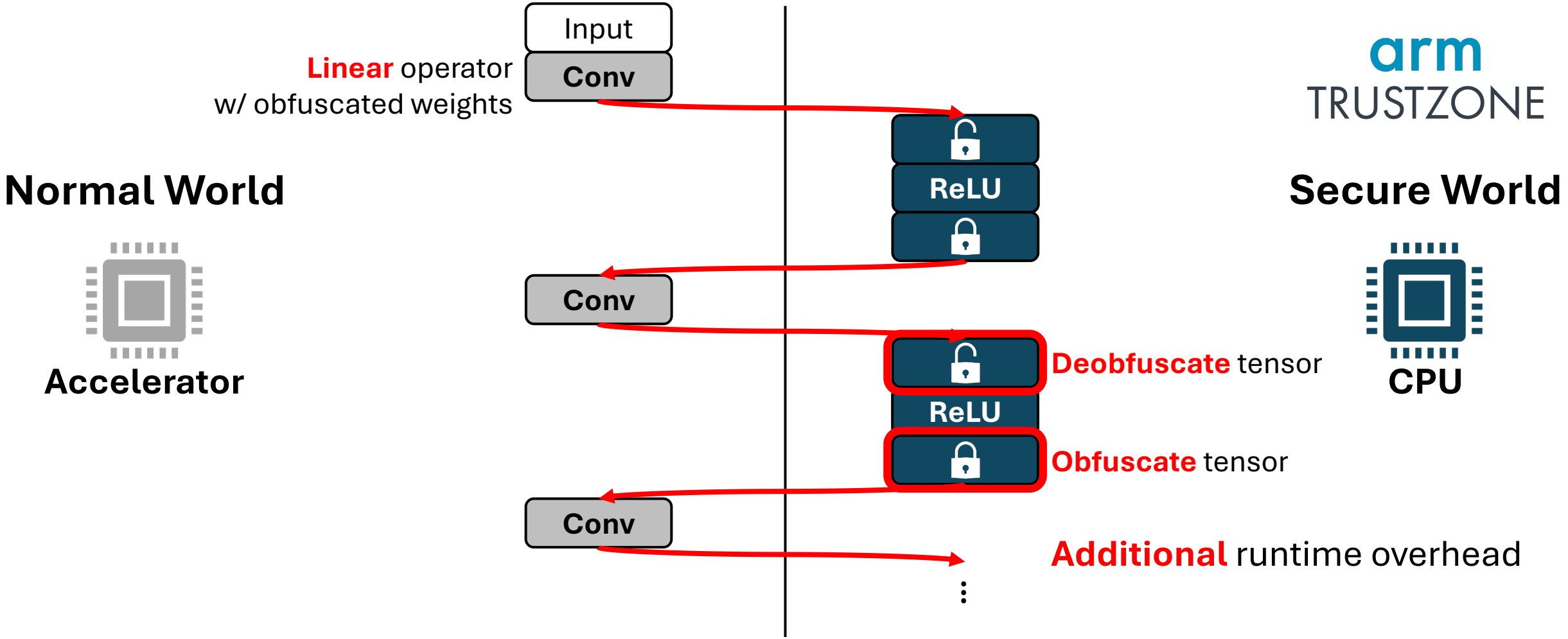
[4] Mo et al., "DarkneTZ: Towards model privacy at the edge using trusted execution environments," *MobiSys '20*

Prior Approach #2: Assigning Accelerator to Normal World



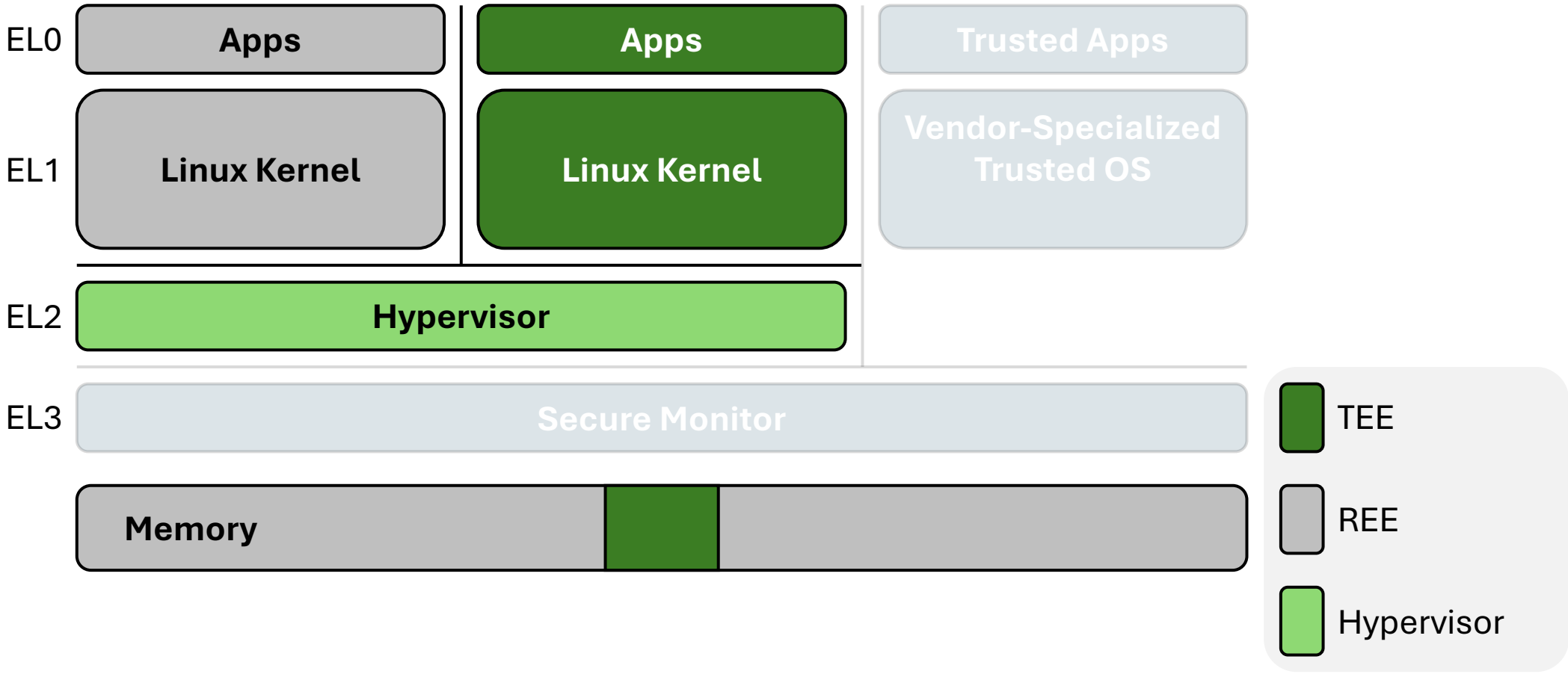
[5] Sun et al., "ShadowNet: A secure and efficient on-device model inference system for convolutional neural networks," *IEEE S&P '23*

Prior Approach #2: Assigning Accelerator to Normal World

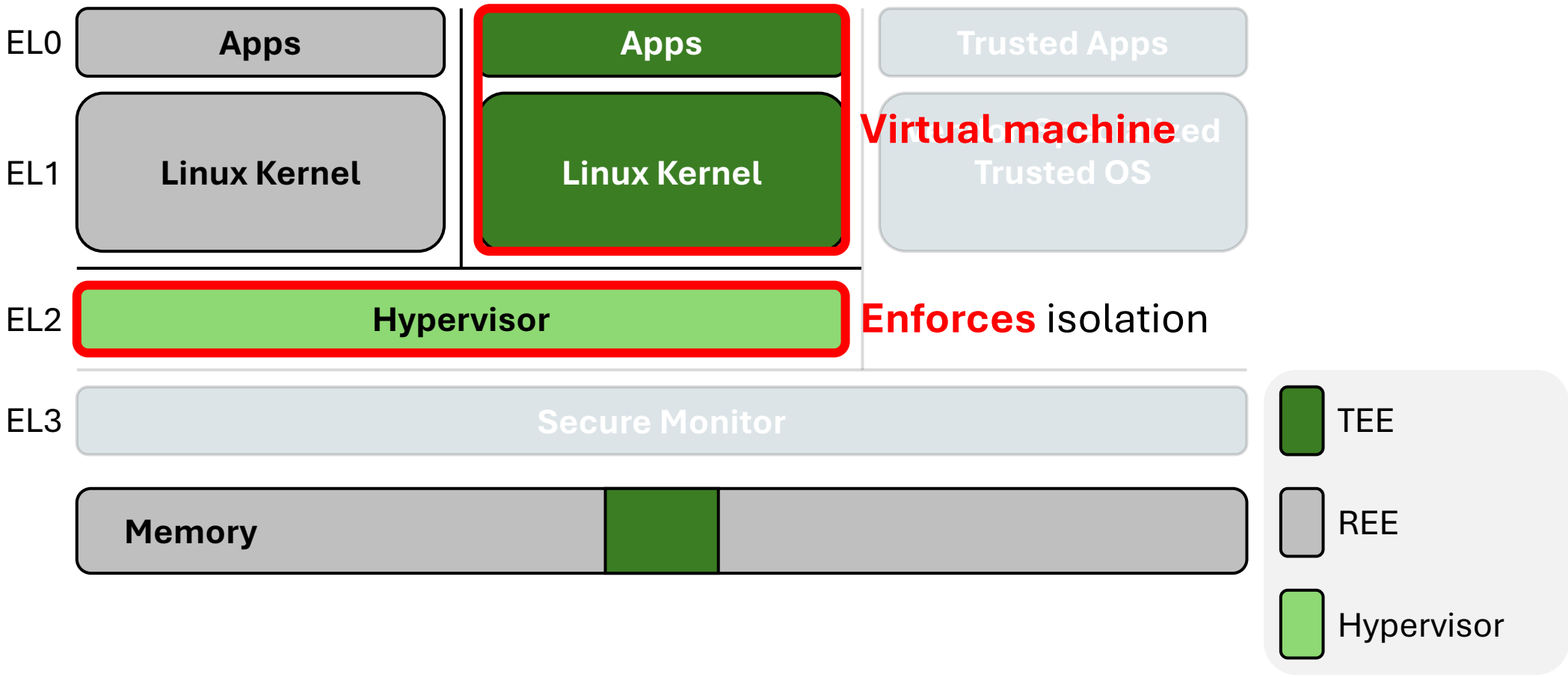


[5] Sun et al., "ShadowNet: A secure and efficient on-device model inference system for convolutional neural networks," *IEEE S&P '23*

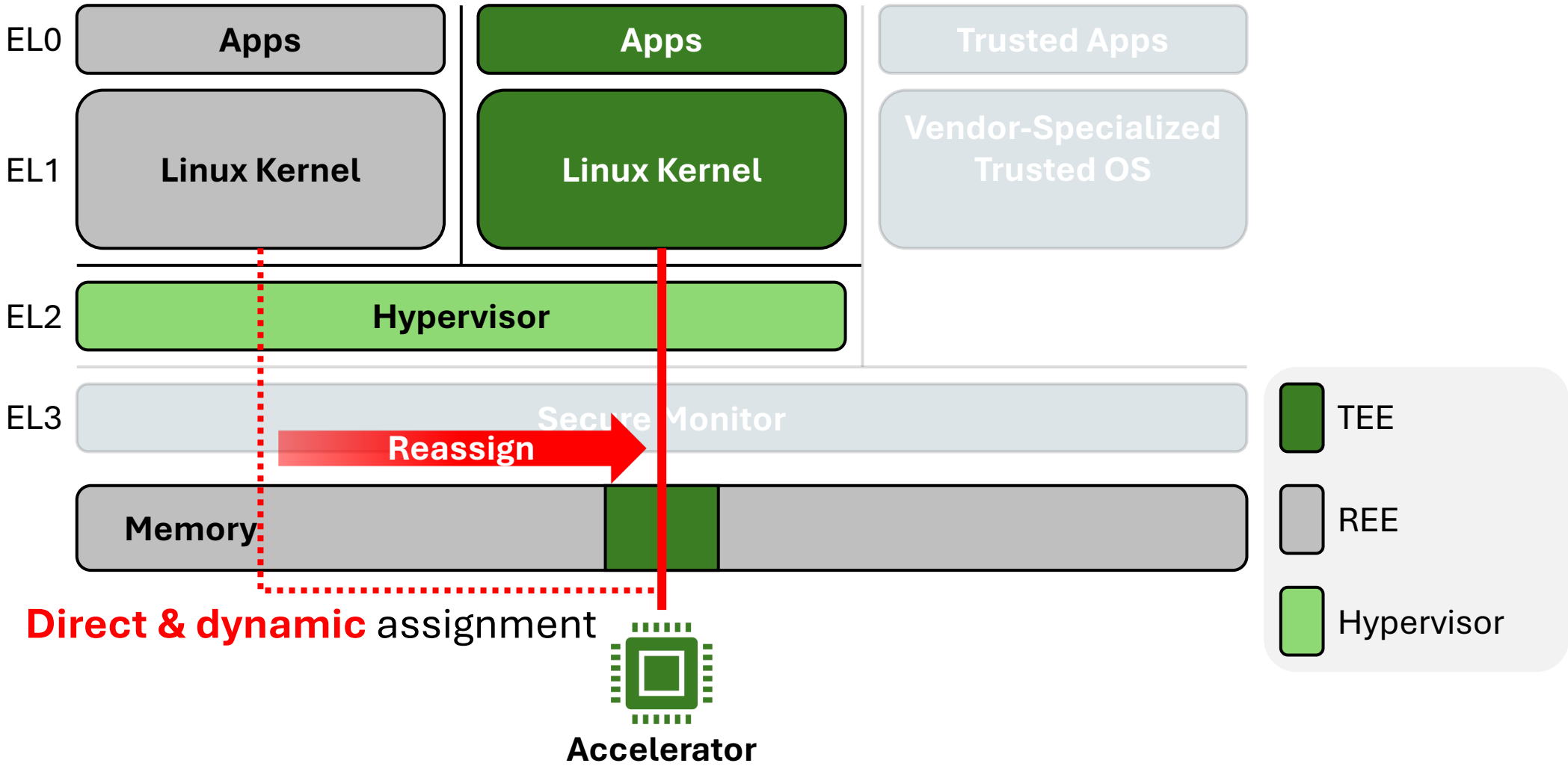
Our Approach: Protection with Virtualization-Based TEE



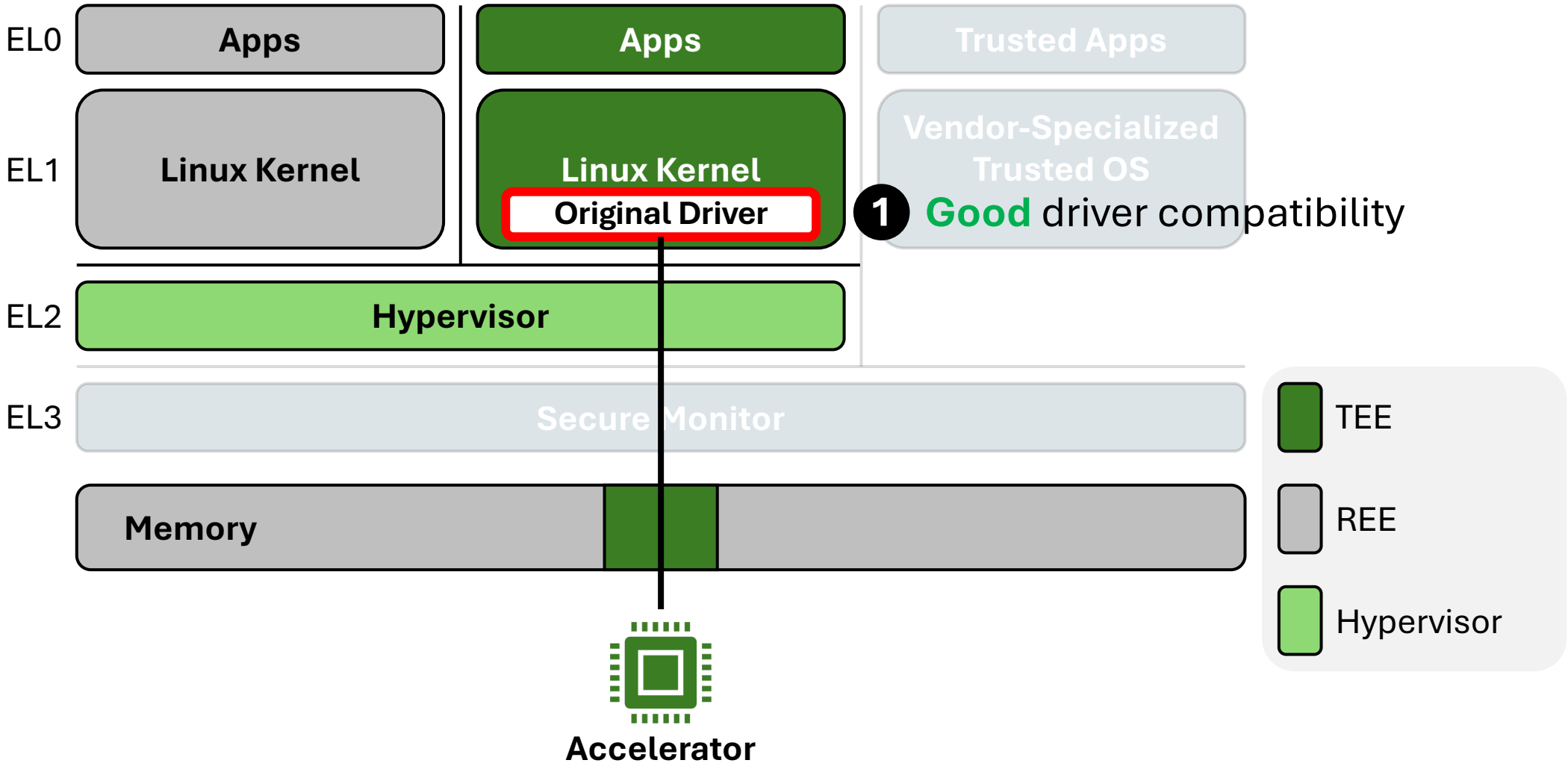
Our Approach: Protection with Virtualization-Based TEE



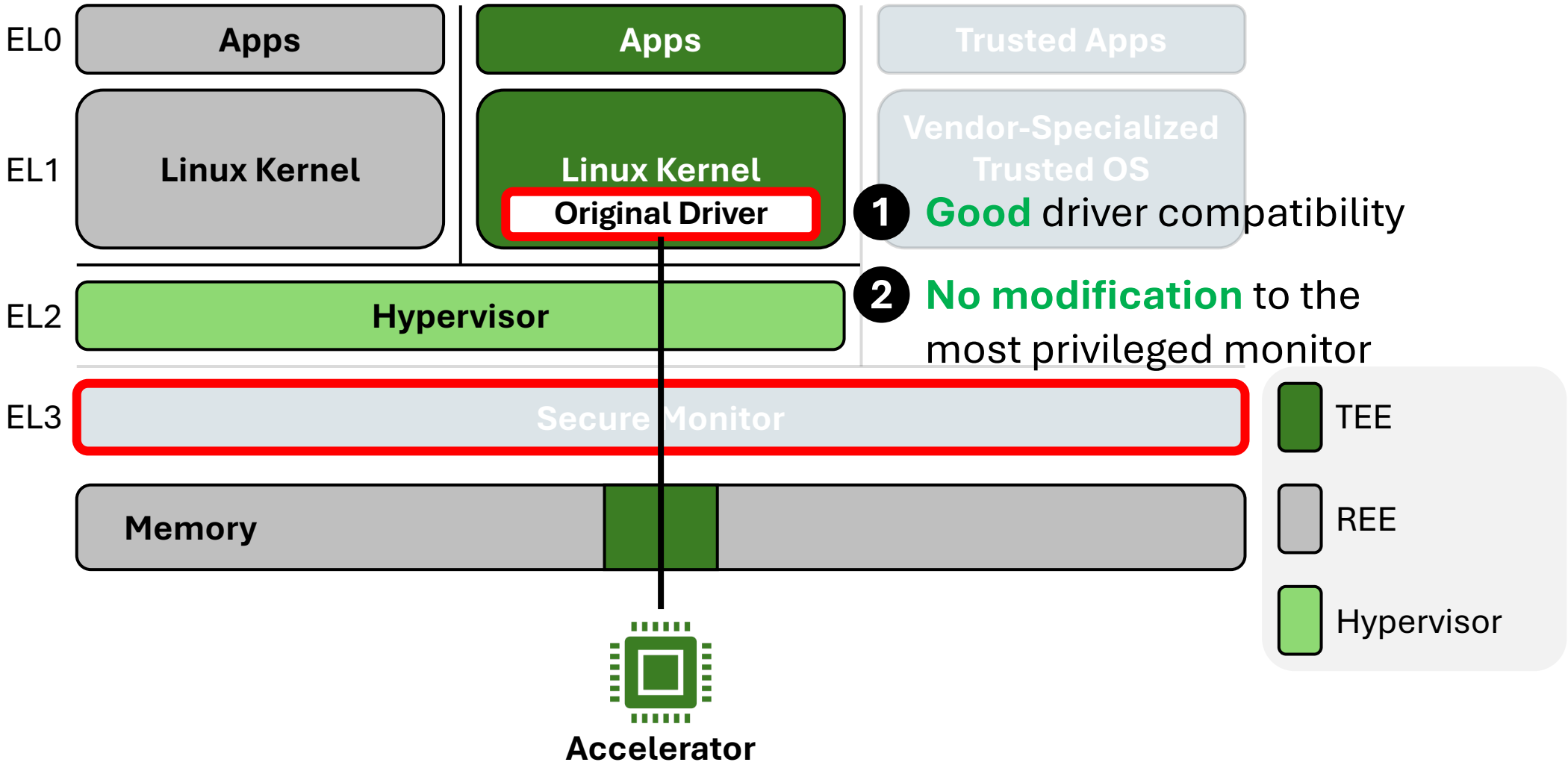
Our Approach: Protection with Virtualization-Based TEE



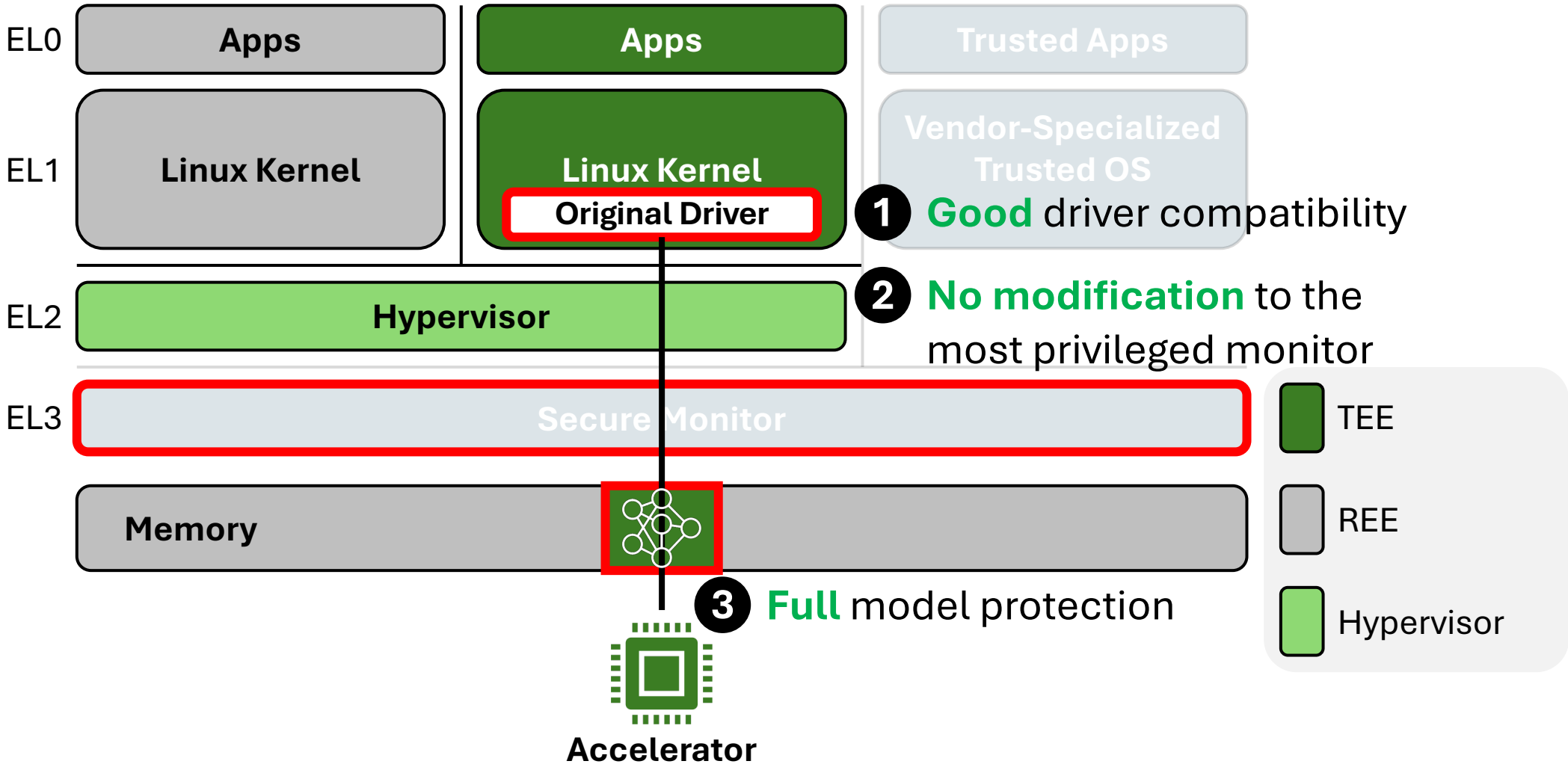
Our Approach: Advantages



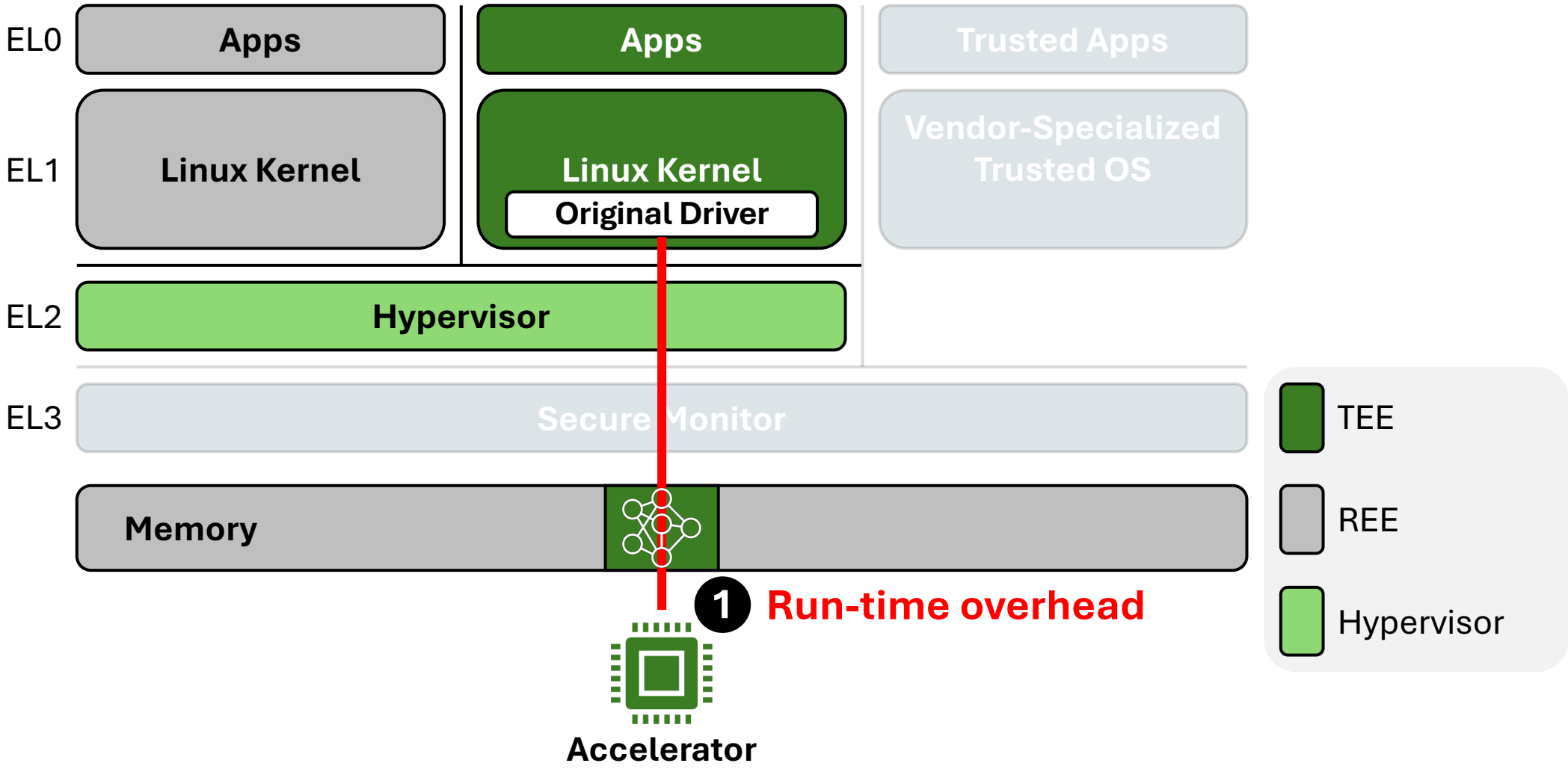
Our Approach: Advantages



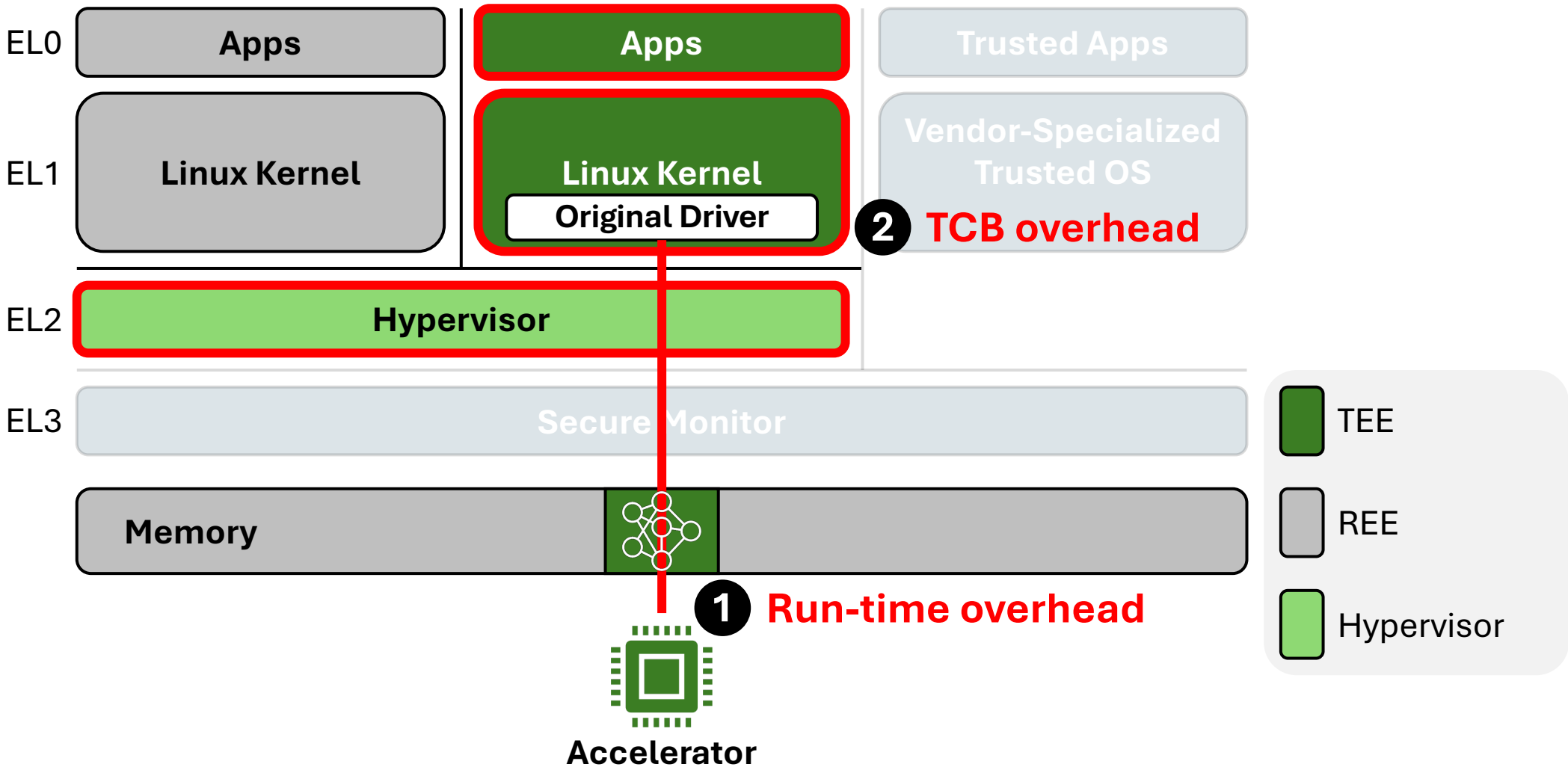
Our Approach: Advantages



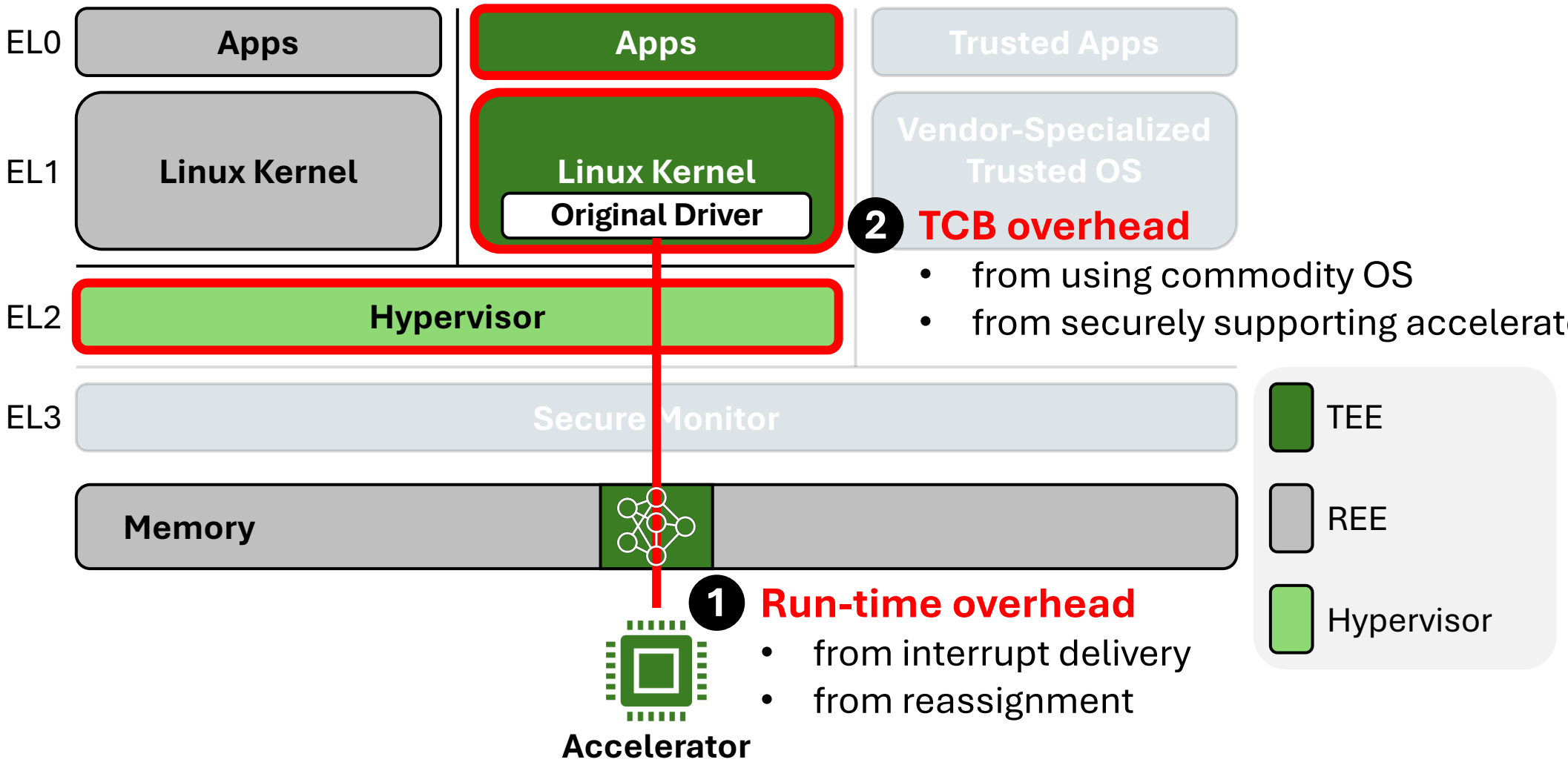
Our Approach: Challenges



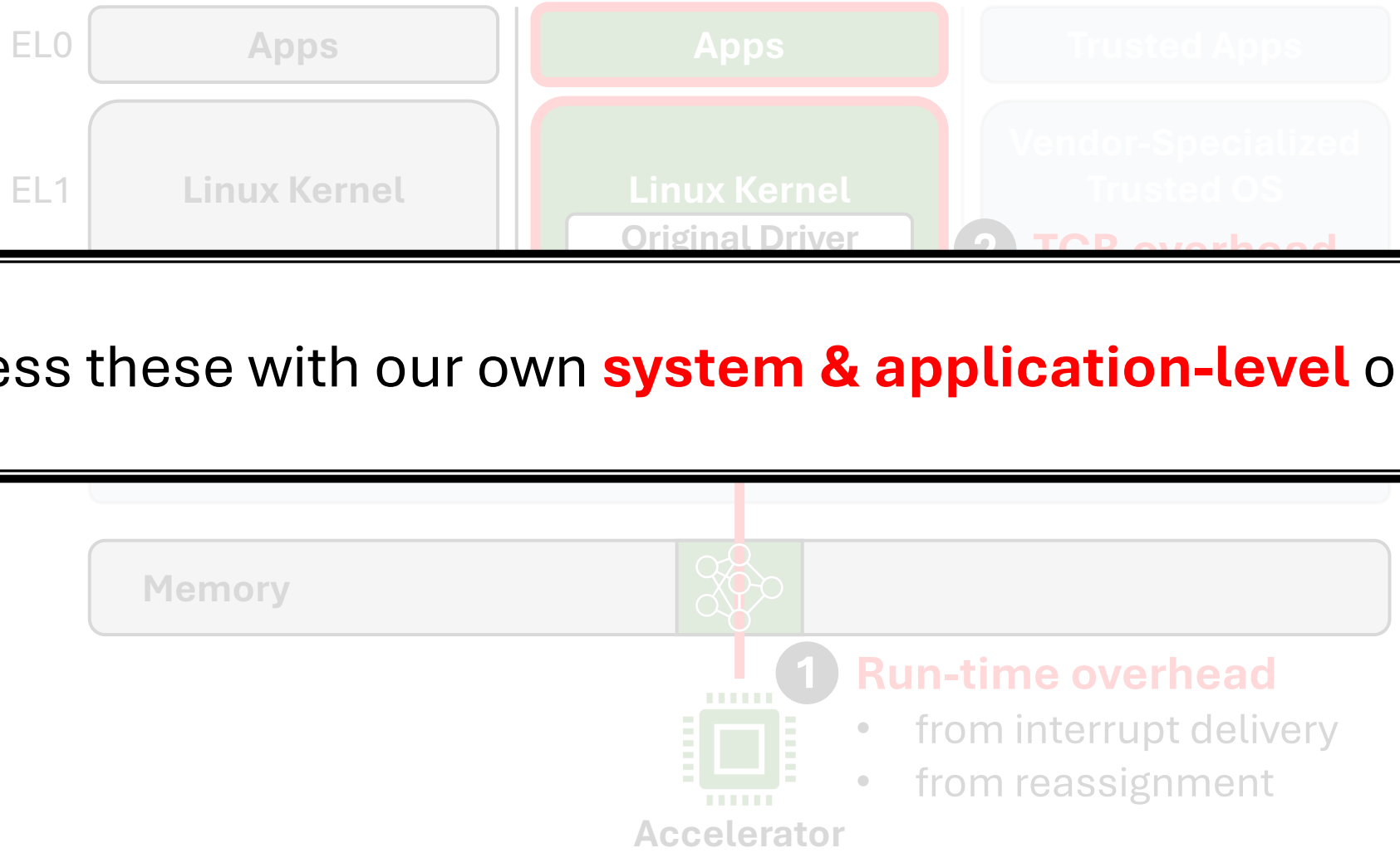
Our Approach: Challenges



Our Approach: Challenges

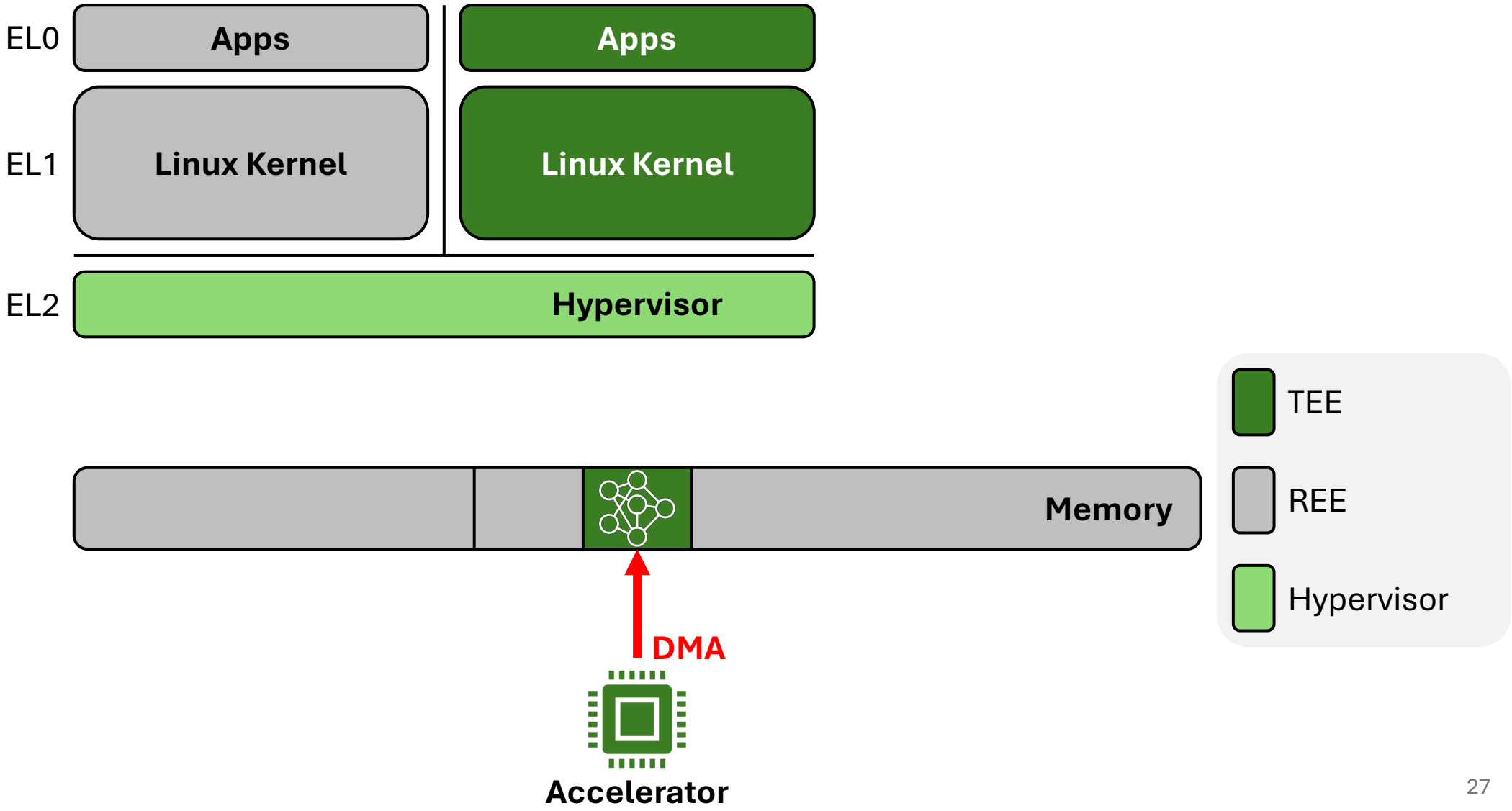


Our Approach: Challenges

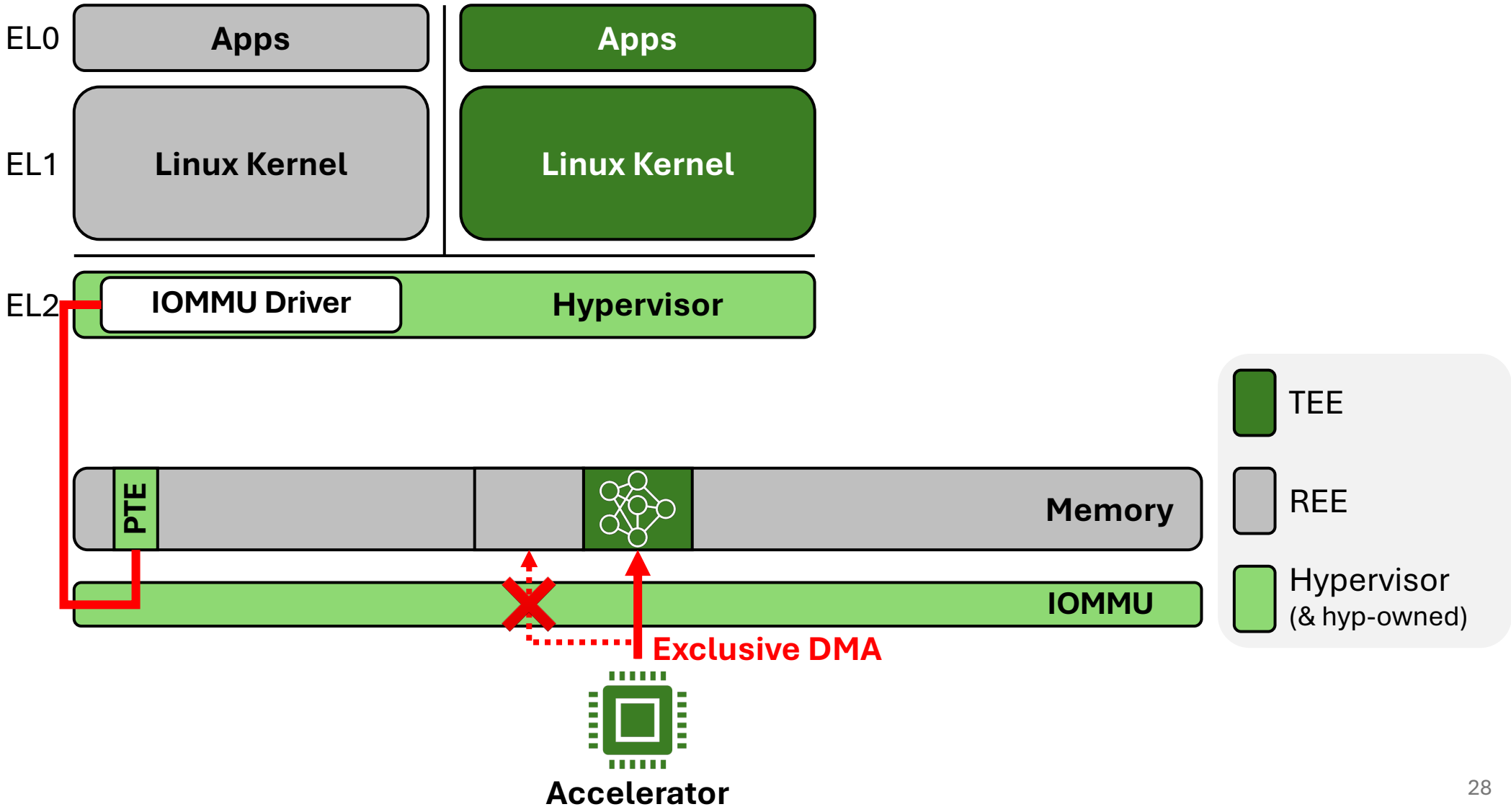


We address these with our own **system & application-level** optimizations!

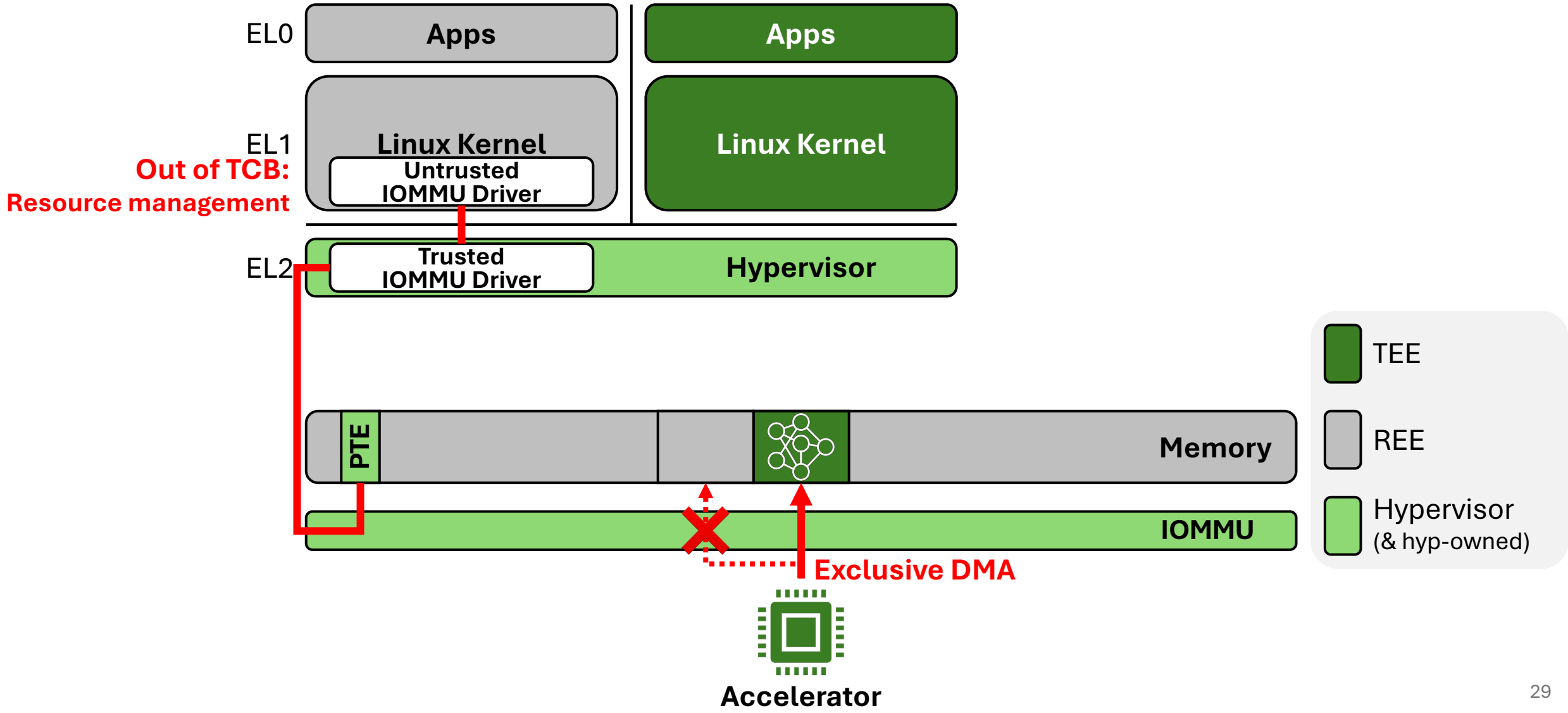
System-level Optimization: Fast & Secure Reassignment



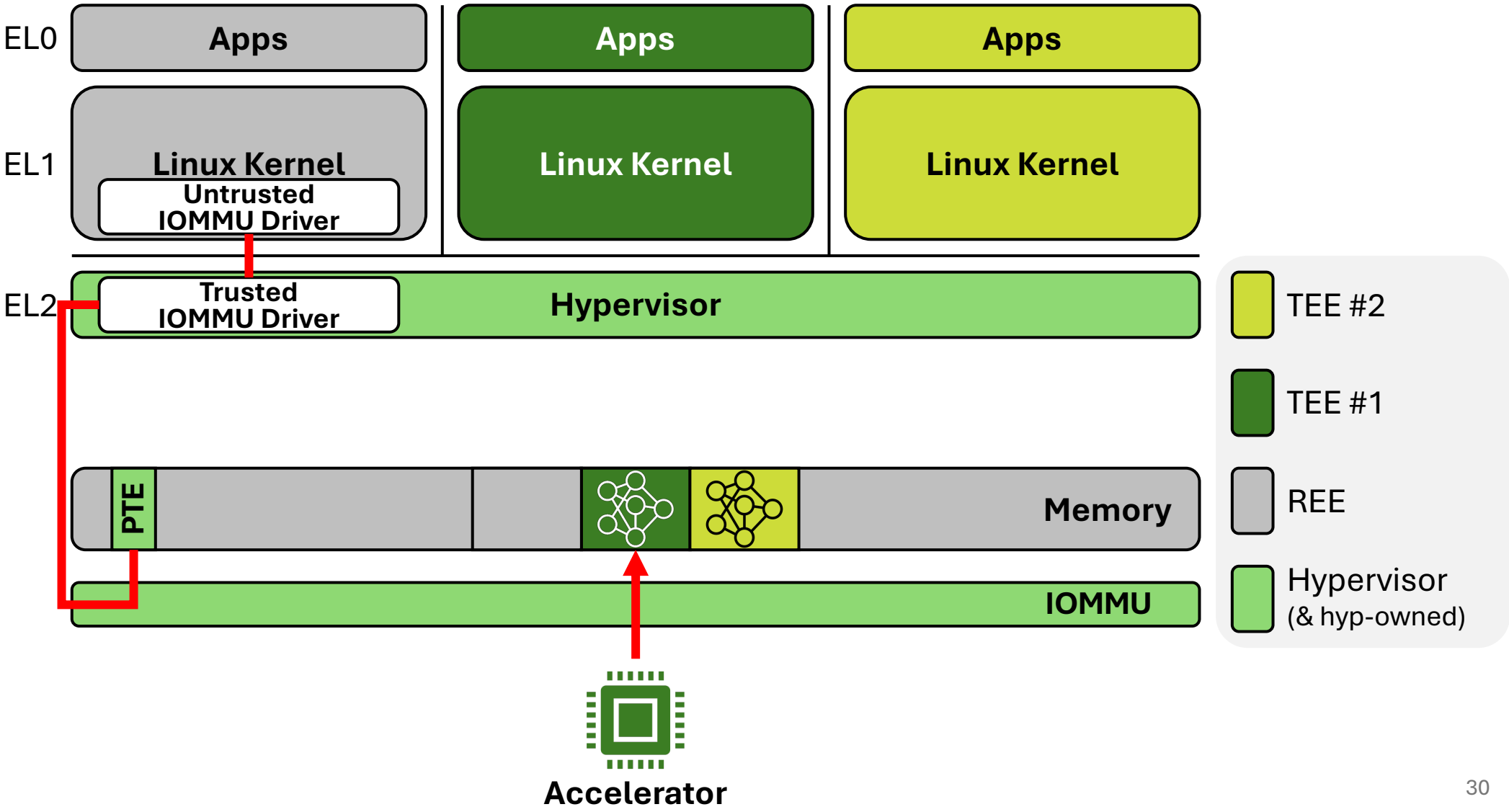
System-level Optimization: Fast & Secure Reassignment



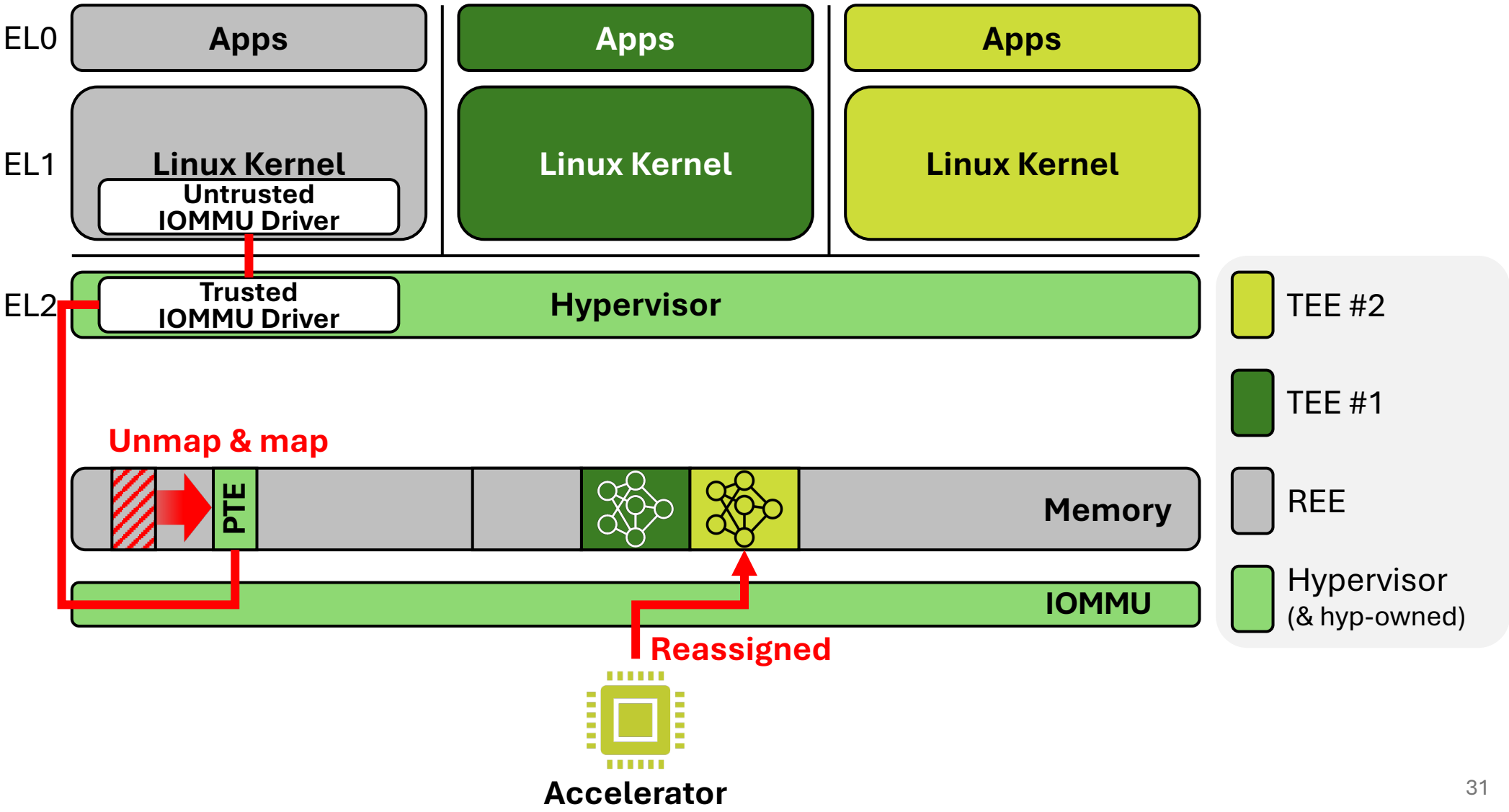
System-level Optimization: Fast & Secure Reassignment



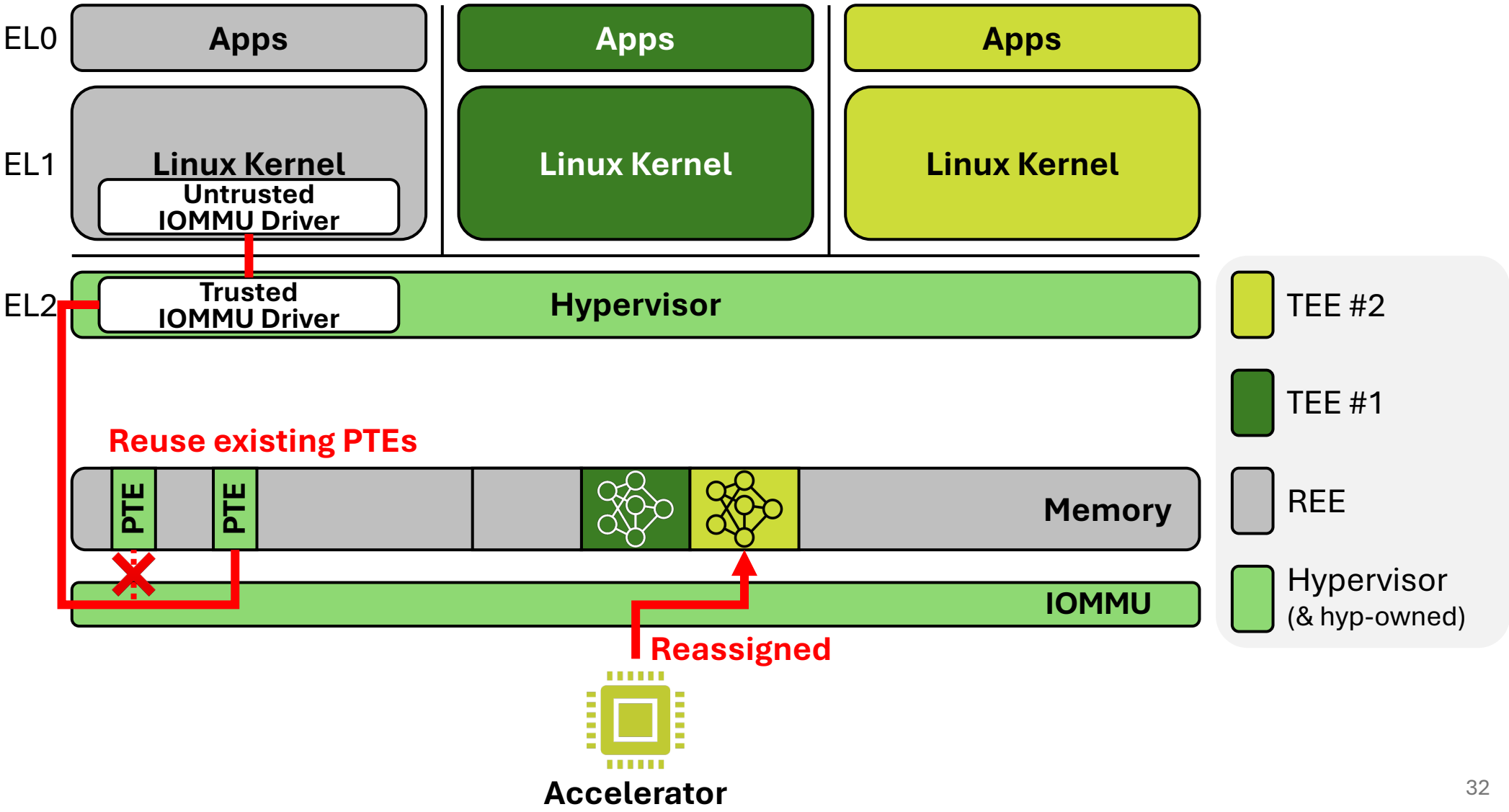
System-level Optimization: Fast & Secure Reassignment



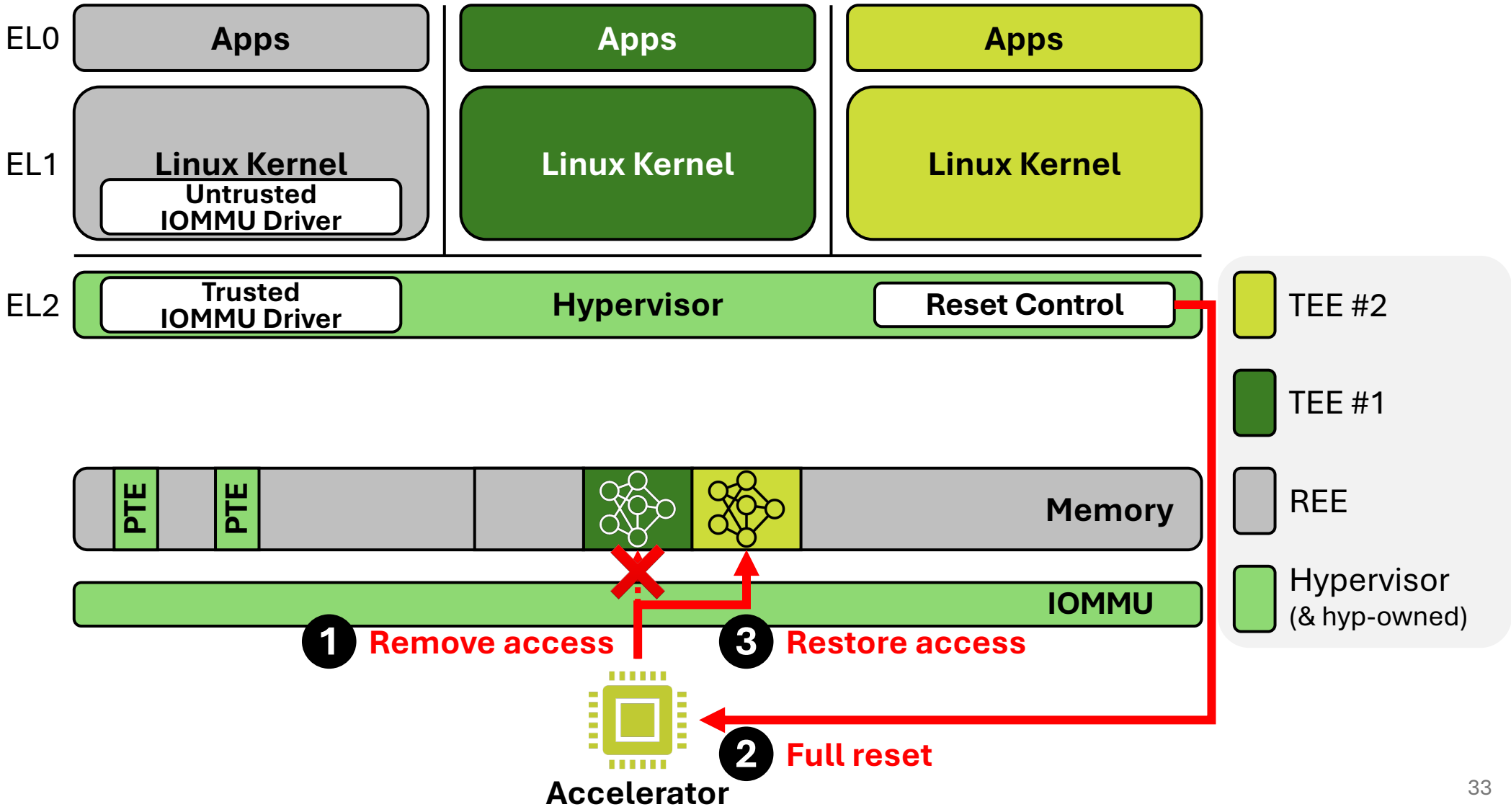
System-level Optimization: Fast & Secure Reassignment



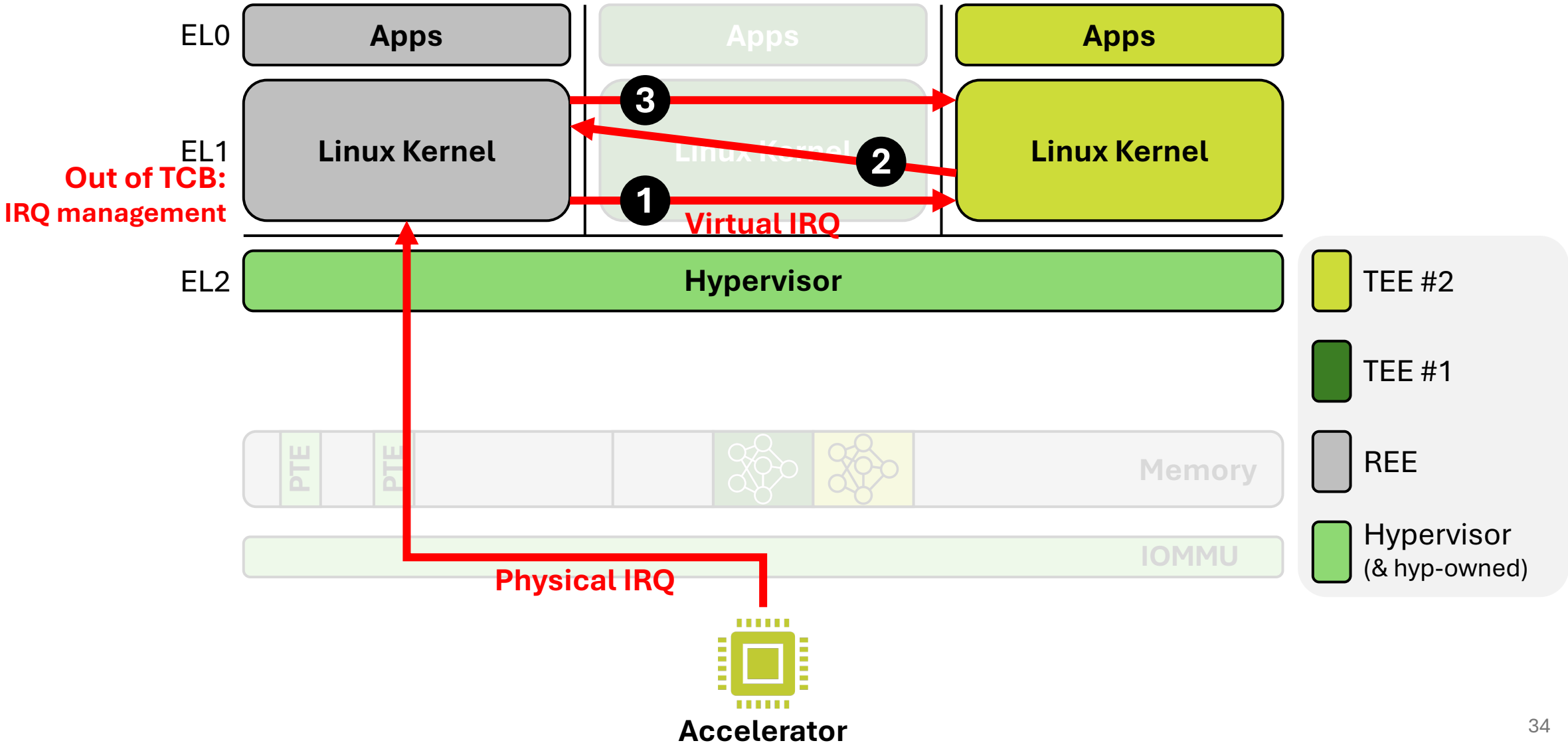
System-level Optimization: Fast & Secure Reassignment



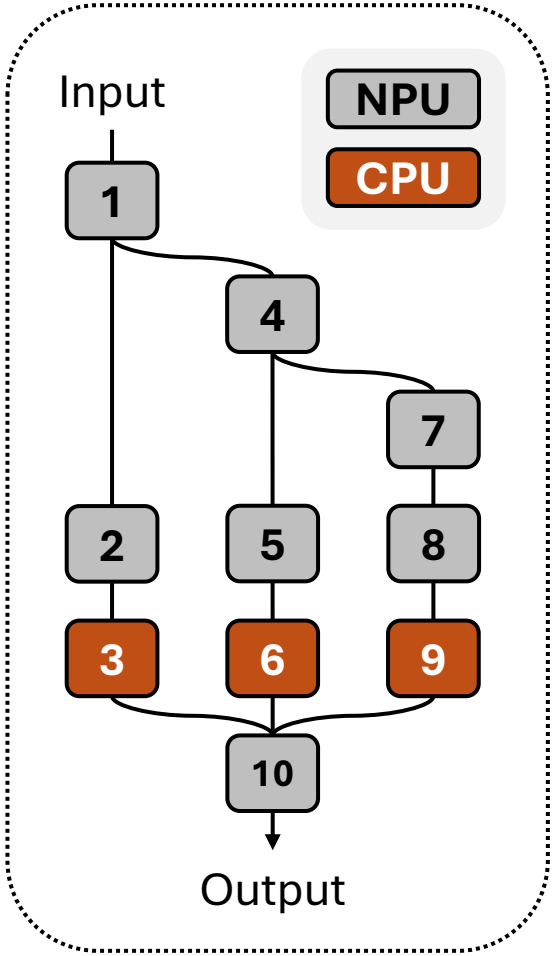
System-level Optimization: Fast & Secure Reassignment



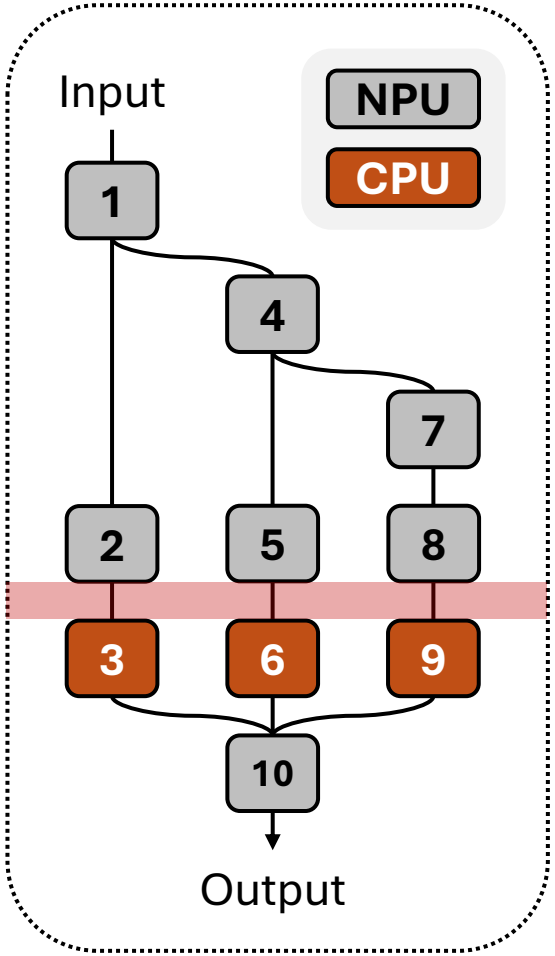
Challenge: Run-time Overhead from Interrupt Delivery



App-level Optimization: Exit-Coalescing Execution Planning

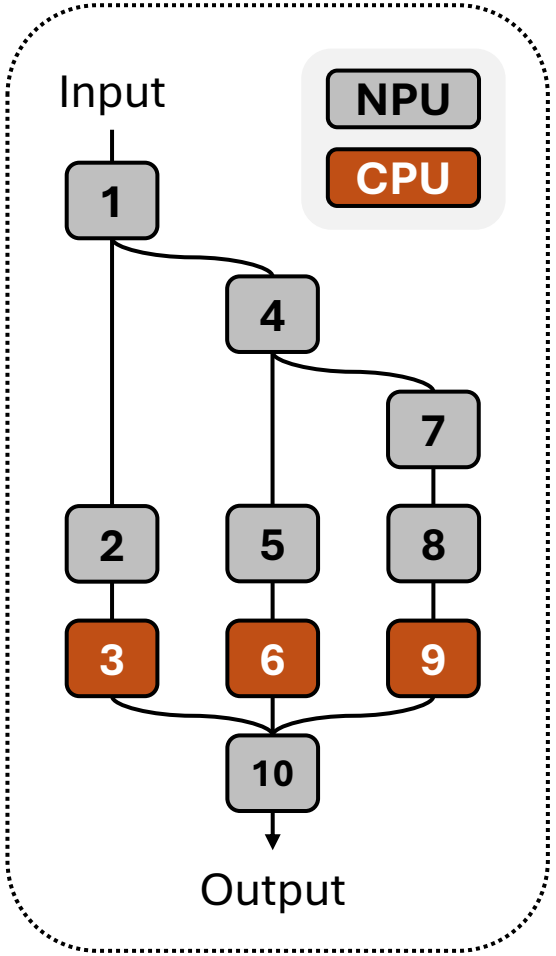


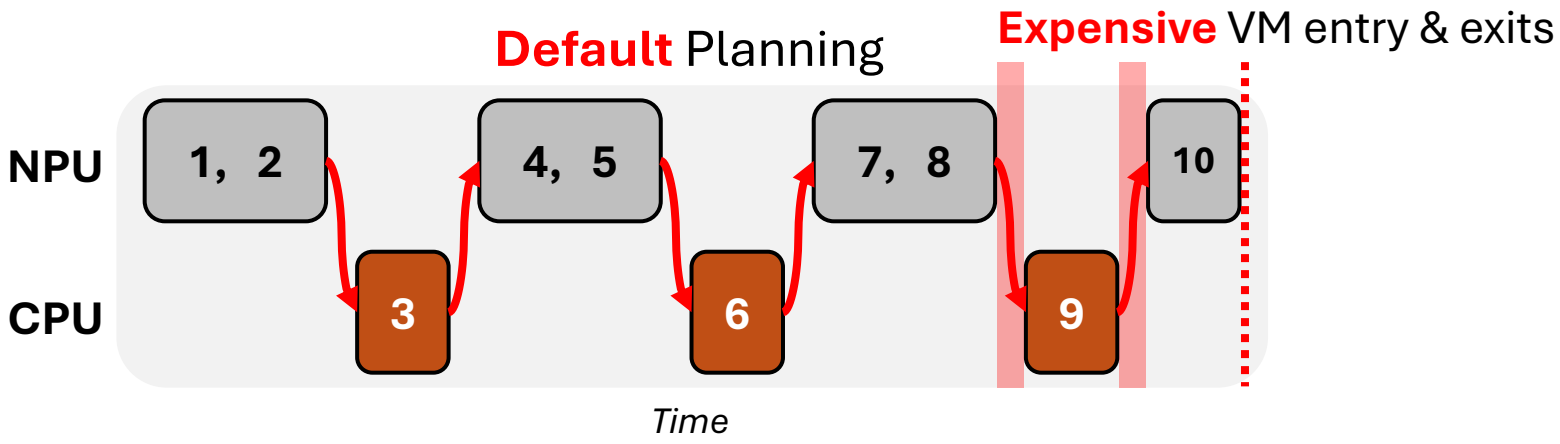
App-level Optimization: Exit-Coalescing Execution Planning



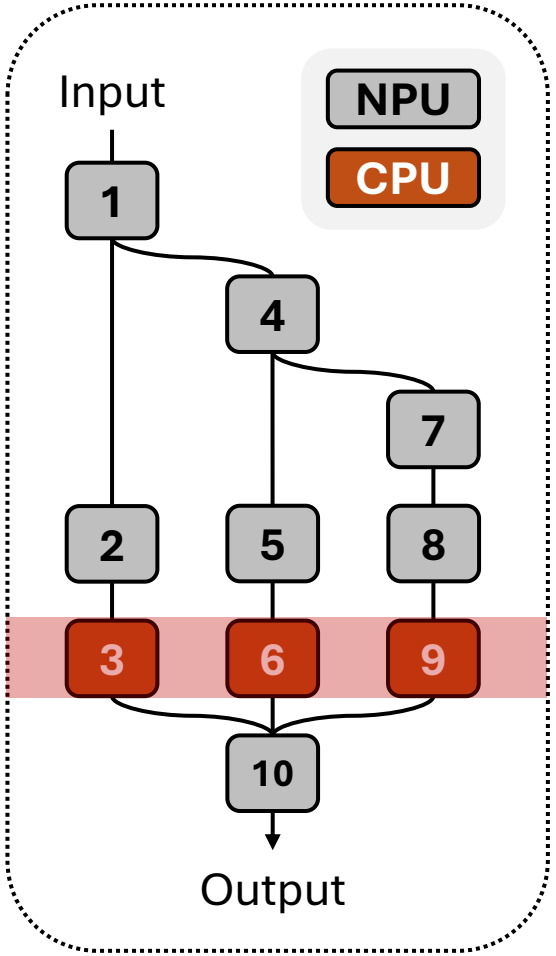
IRQs are delivered

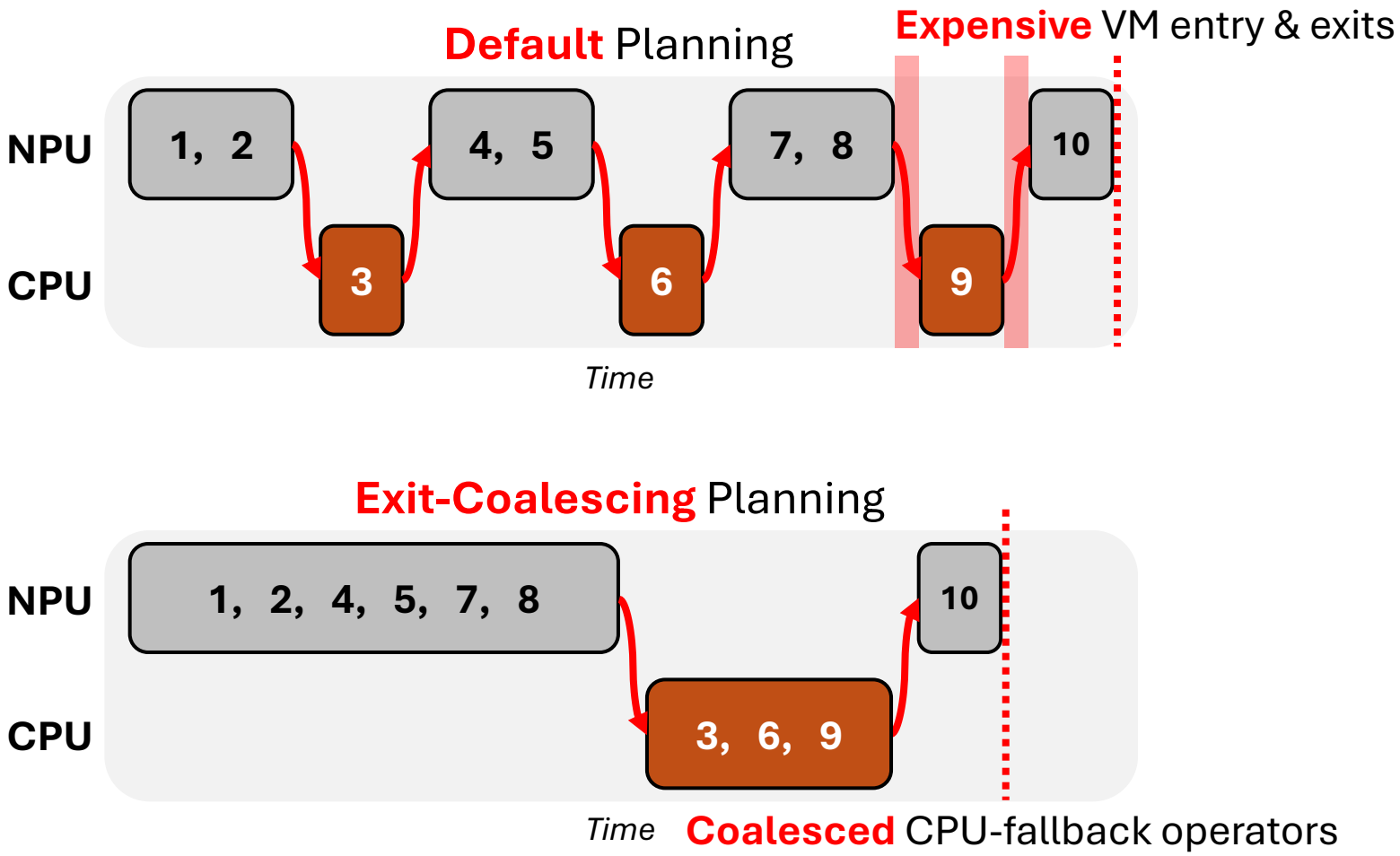
App-level Optimization: Exit-Coalescing Execution Planning



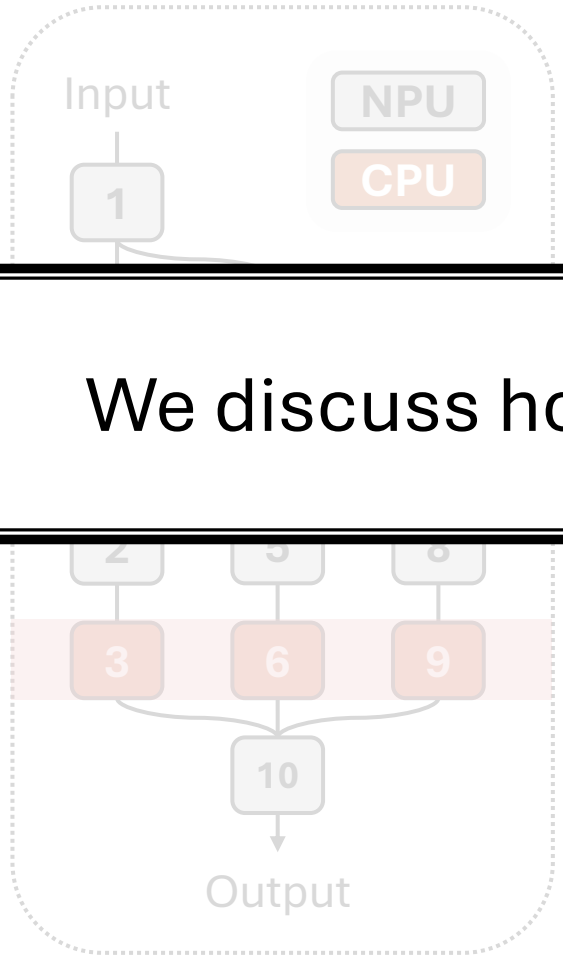


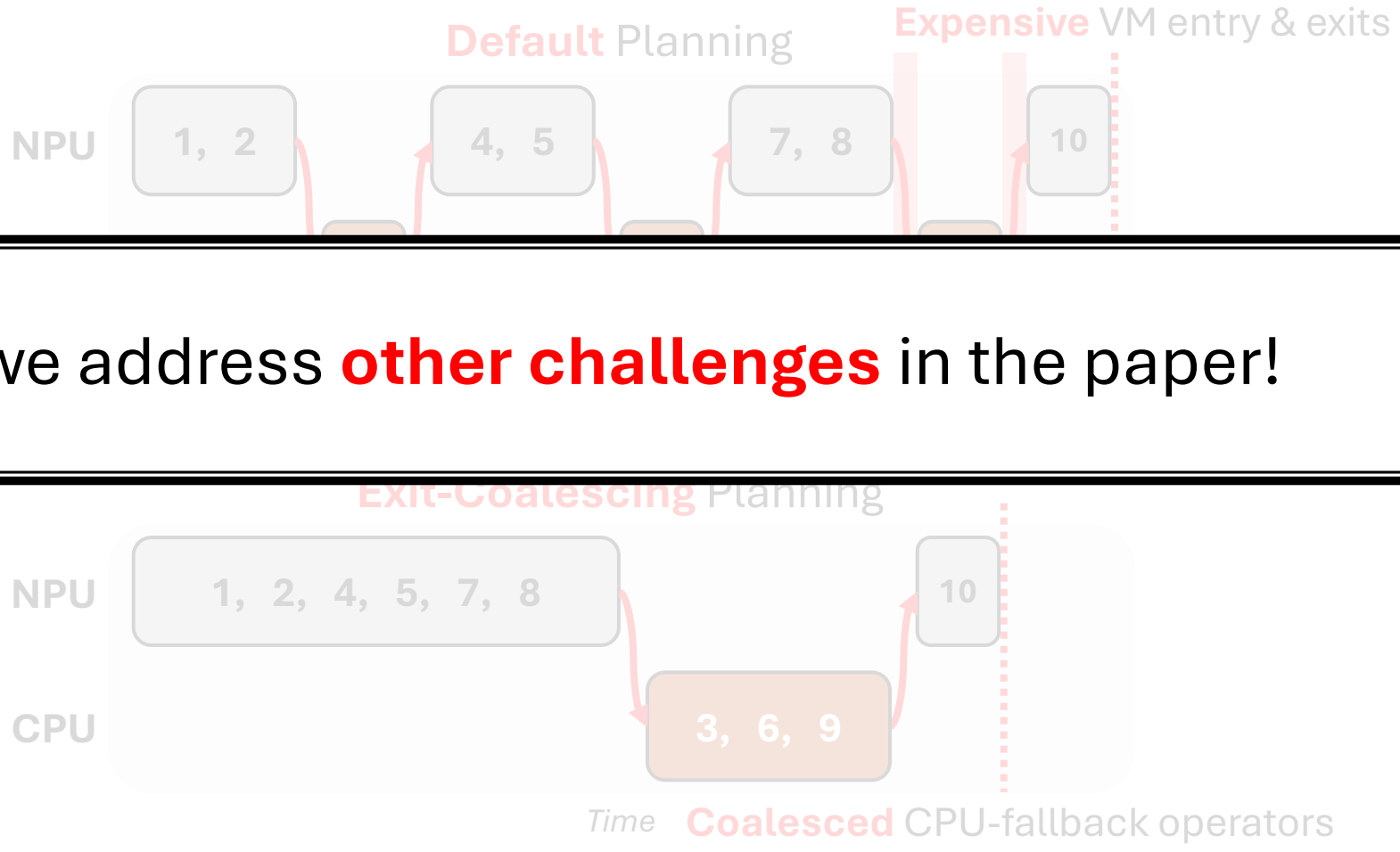
App-level Optimization: Exit-Coalescing Execution Planning





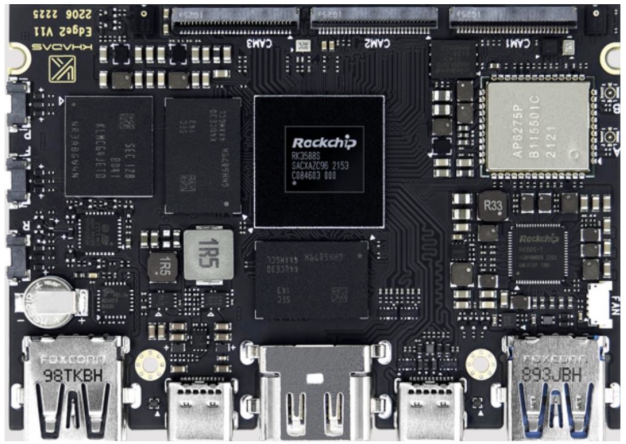
Solution #2: Exit-Coalescing DNN Execution Planning





We discuss how we address **other challenges** in the paper!

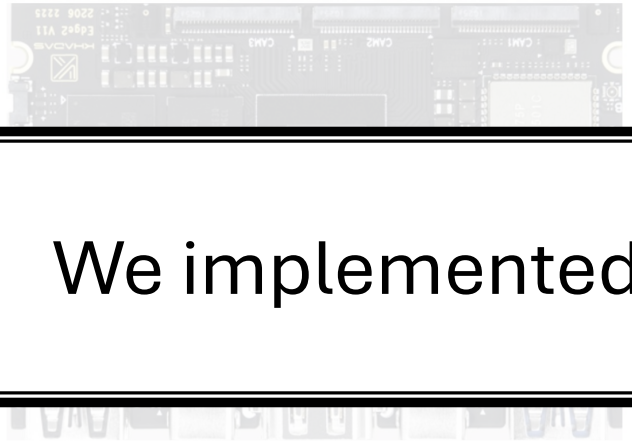
Prototype Implementation



Armv8.2-A Legacy SoC w/ Integrated NPU
(RK3588S, 2021)

Android 13 pKVM + Google CROSVM

Prototype Implementation

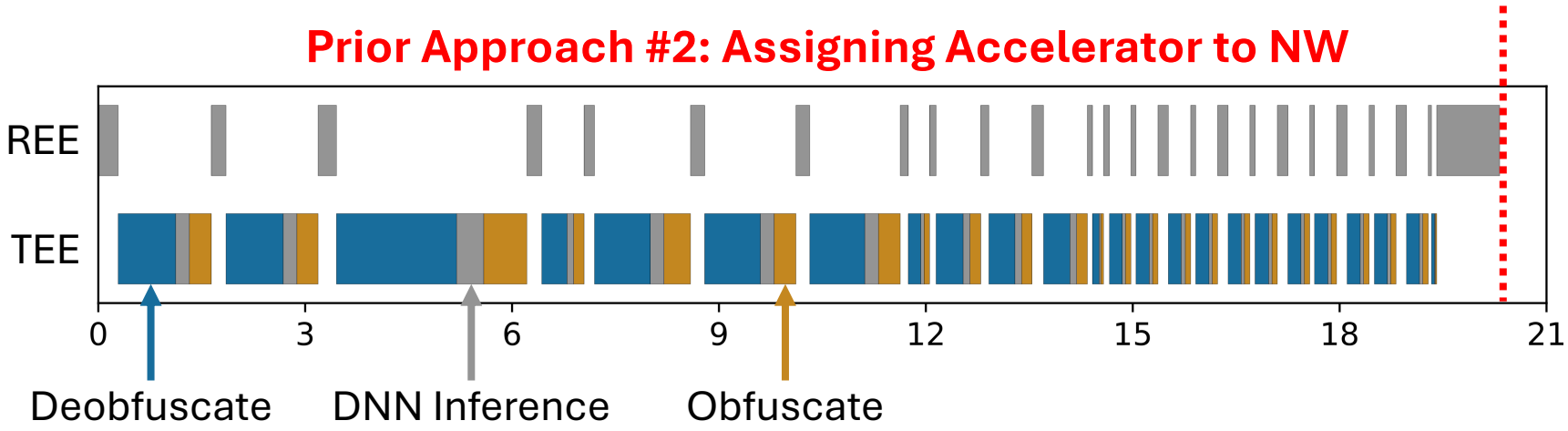


We implemented our prototype on a **legacy SoC**!

Armv8.2-A Legacy SoC w/ Integrated NPU
(RK3588S, 2021)

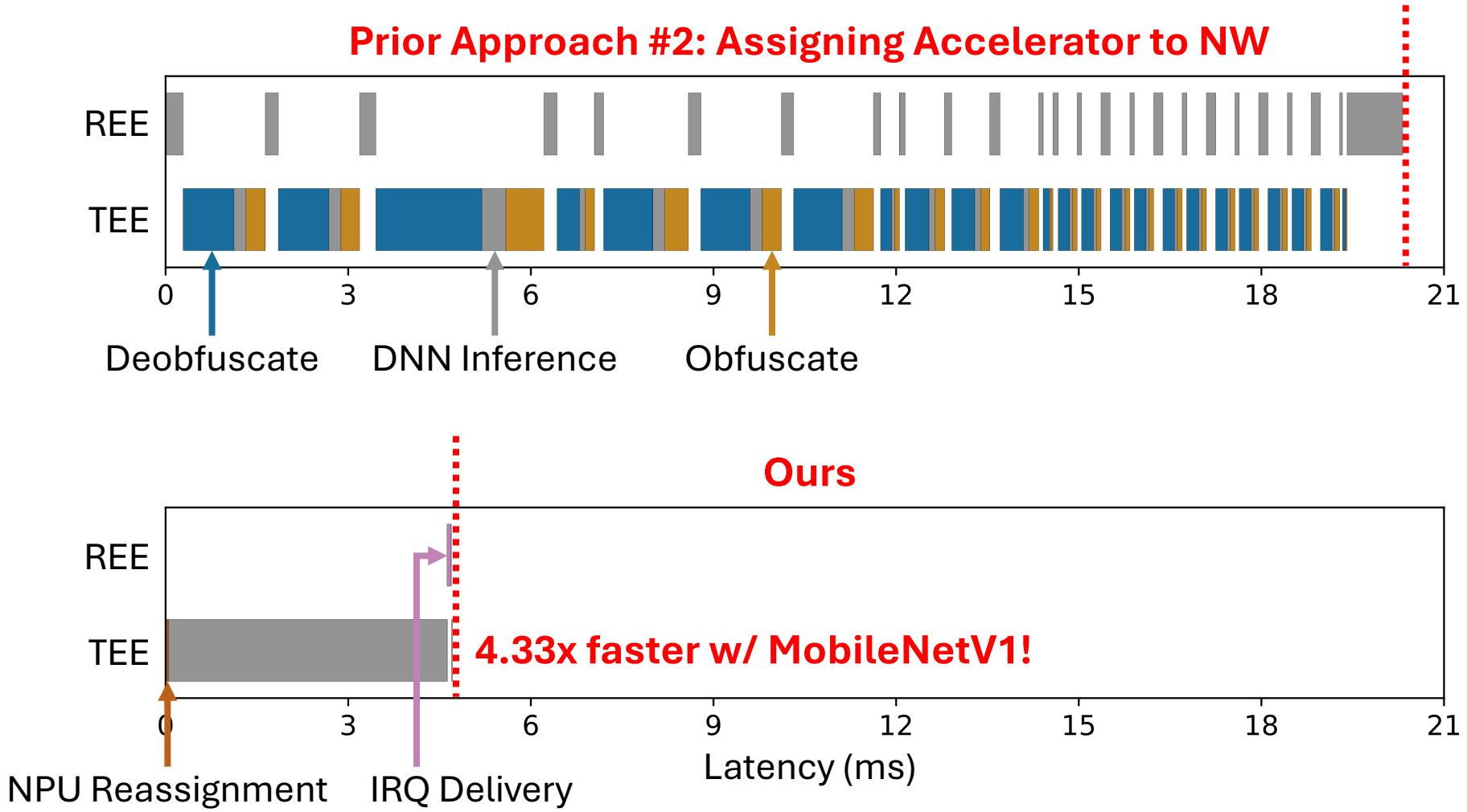
Android 13 pKVM + Google CROSVM

Evaluation #1: DNN Inference Latency with MobileNetV1



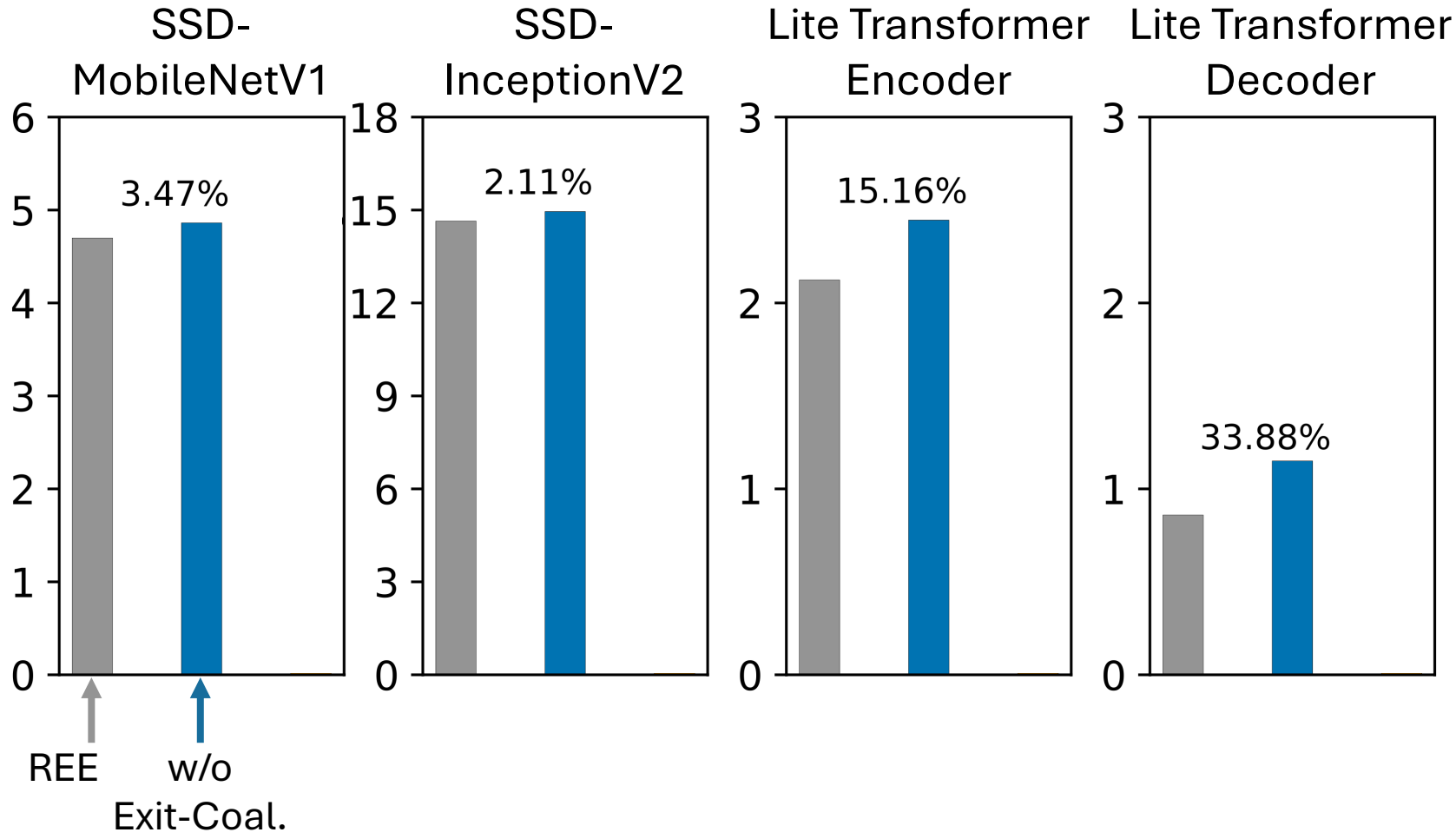
[5] Sun et al., "ShadowNet: A secure and efficient on-device model inference system for convolutional neural networks," *IEEE S&P '23*

Evaluation #1: DNN Inference Latency with MobileNetV1



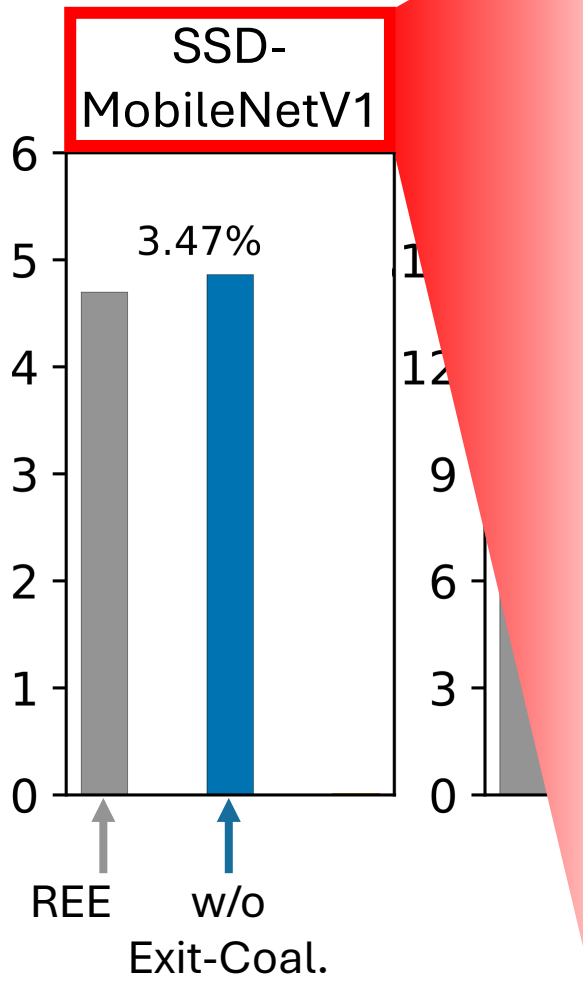
[5] Sun et al., "ShadowNet: A secure and efficient on-device model inference system for convolutional neural networks," *IEEE S&P '23*

Evaluation #2: Exit-Coalescing DNN Execution Planning



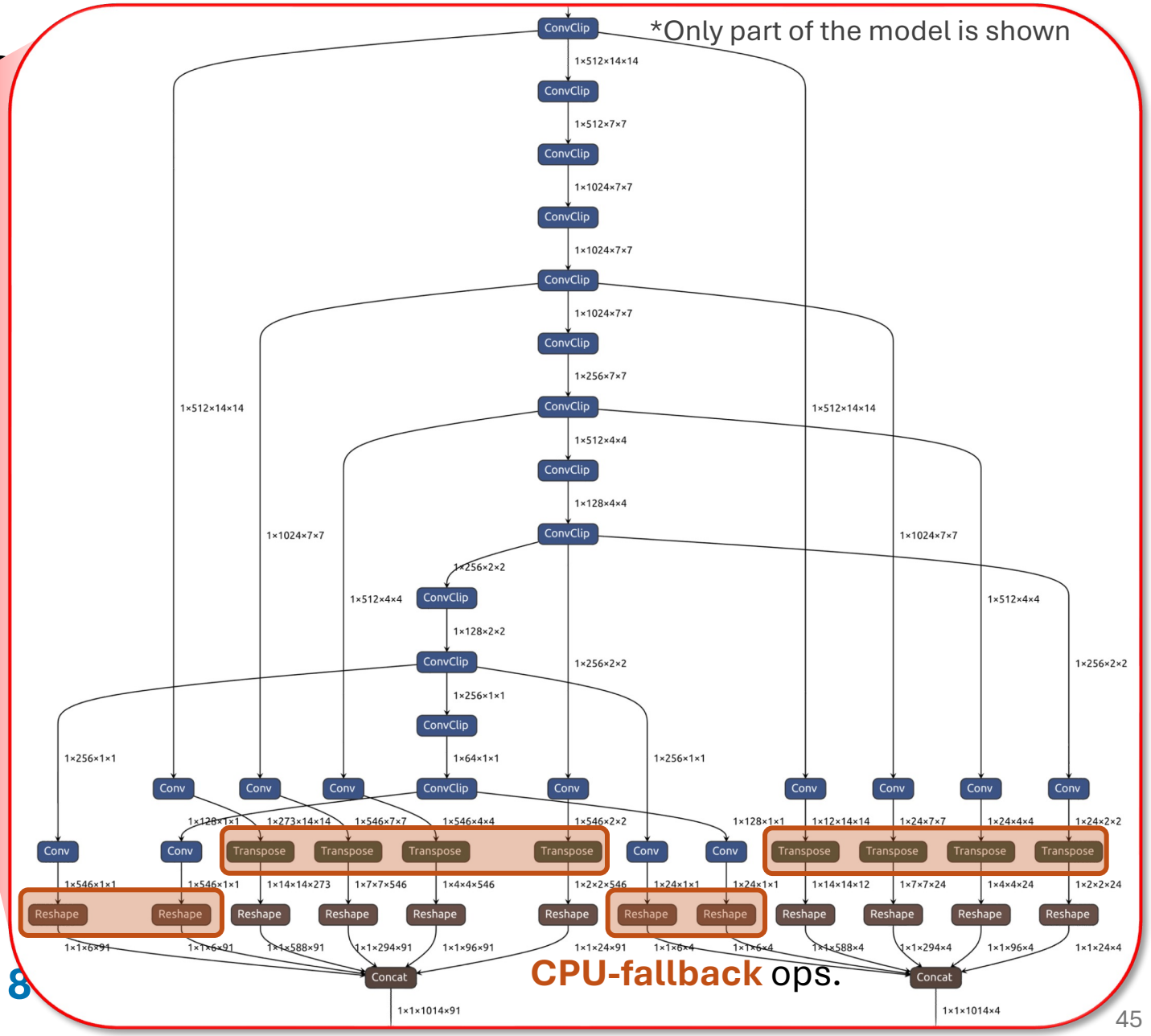
Number of exits: **13** **18** **26** **38**

Evaluation #2: Exit-Coal

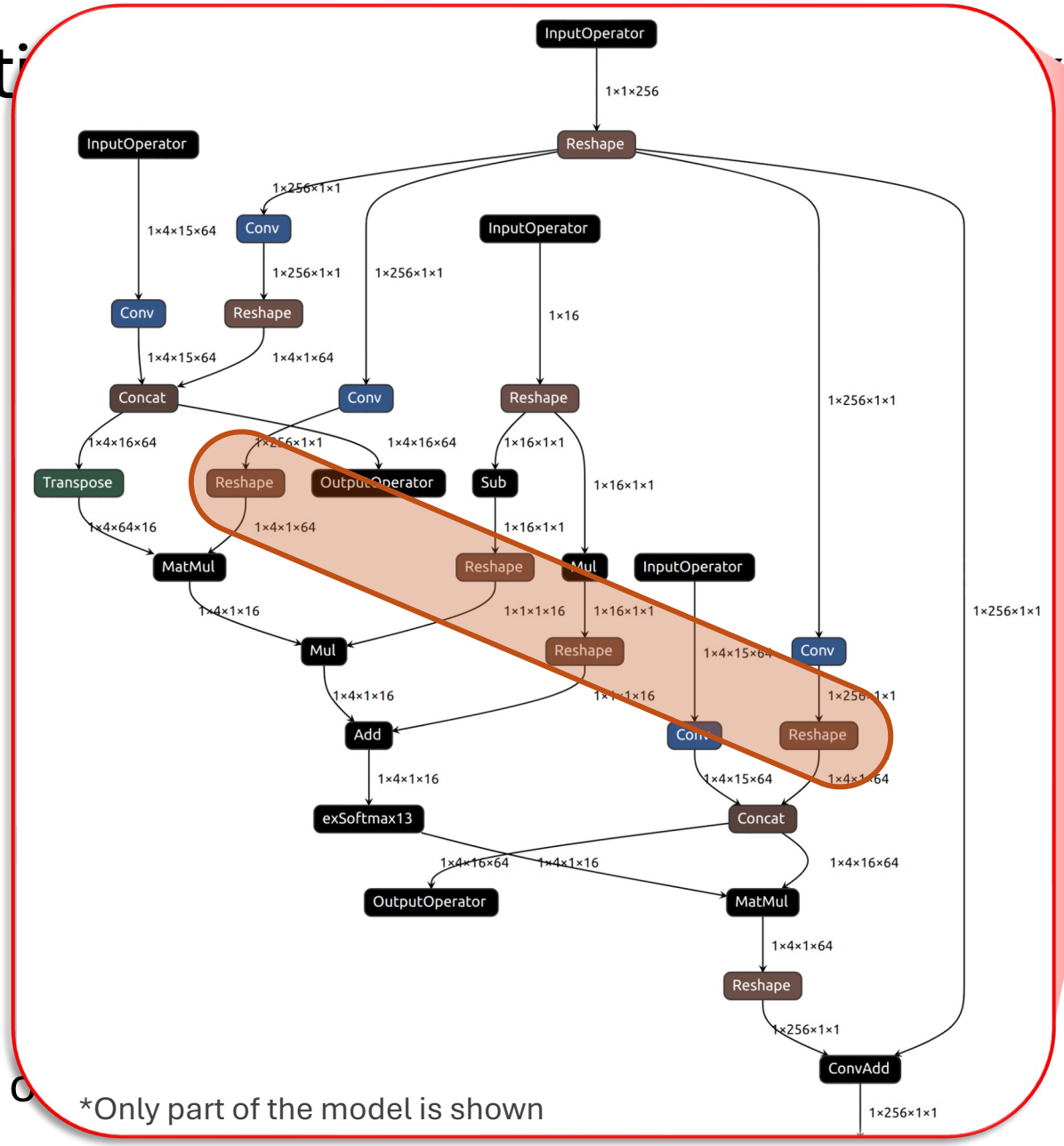


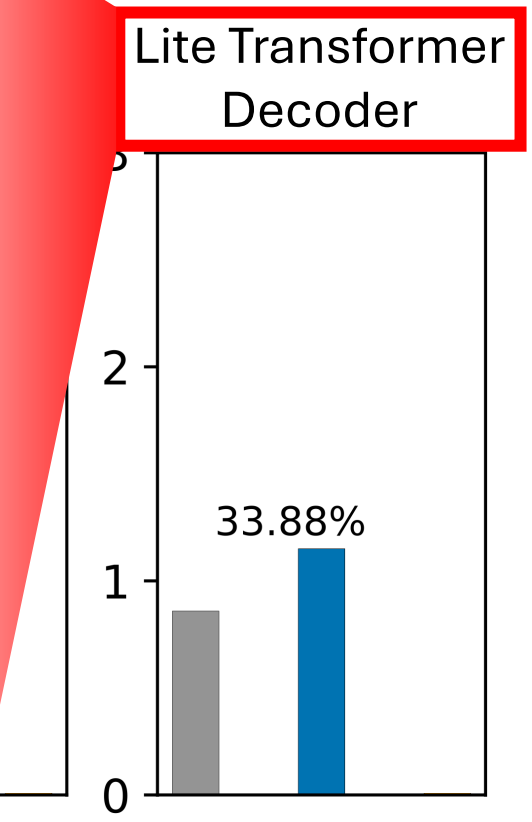
Number of exits: **13**

18



Evaluati ecution Planning



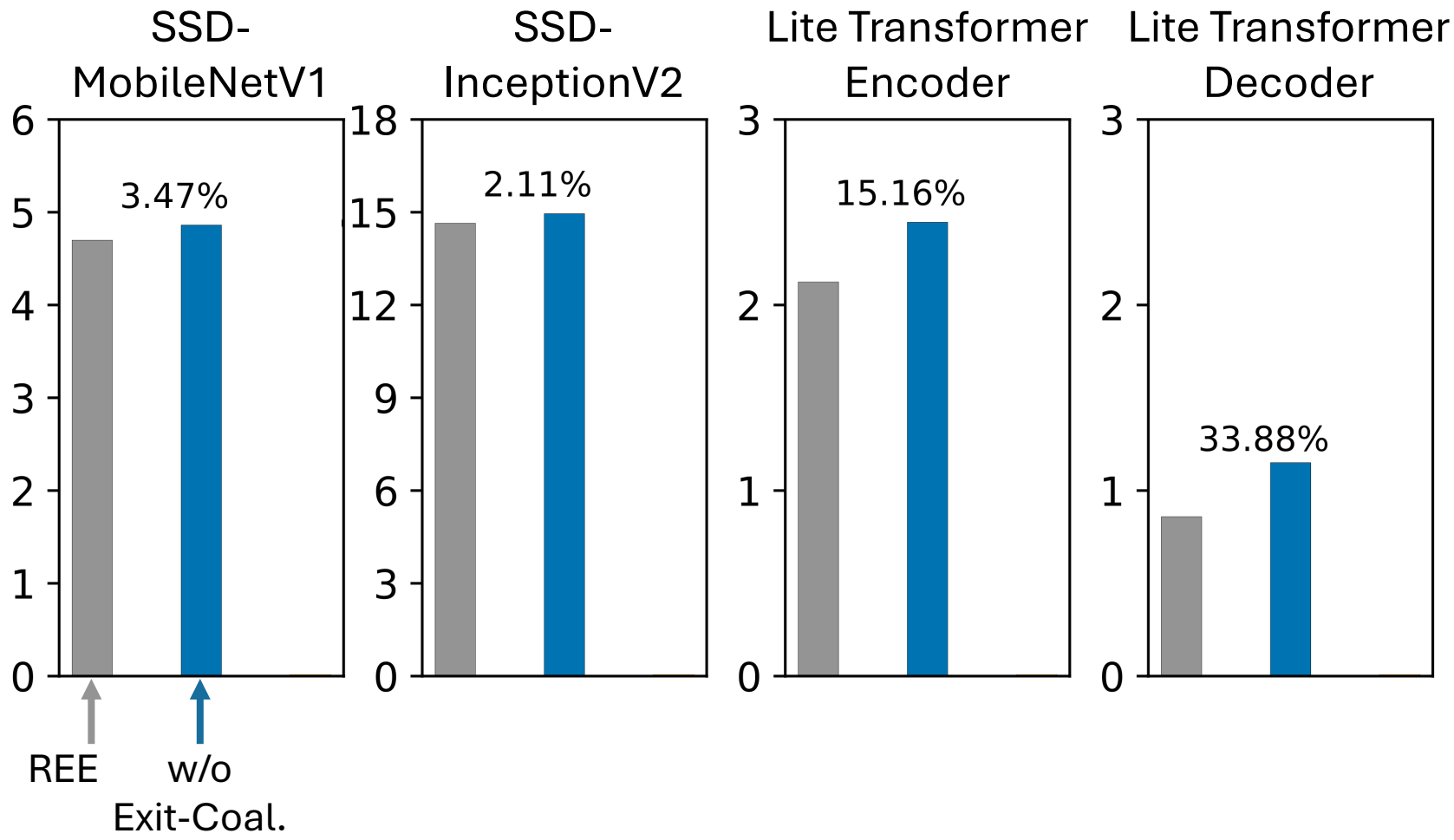


Number o

*Only part of the model is shown

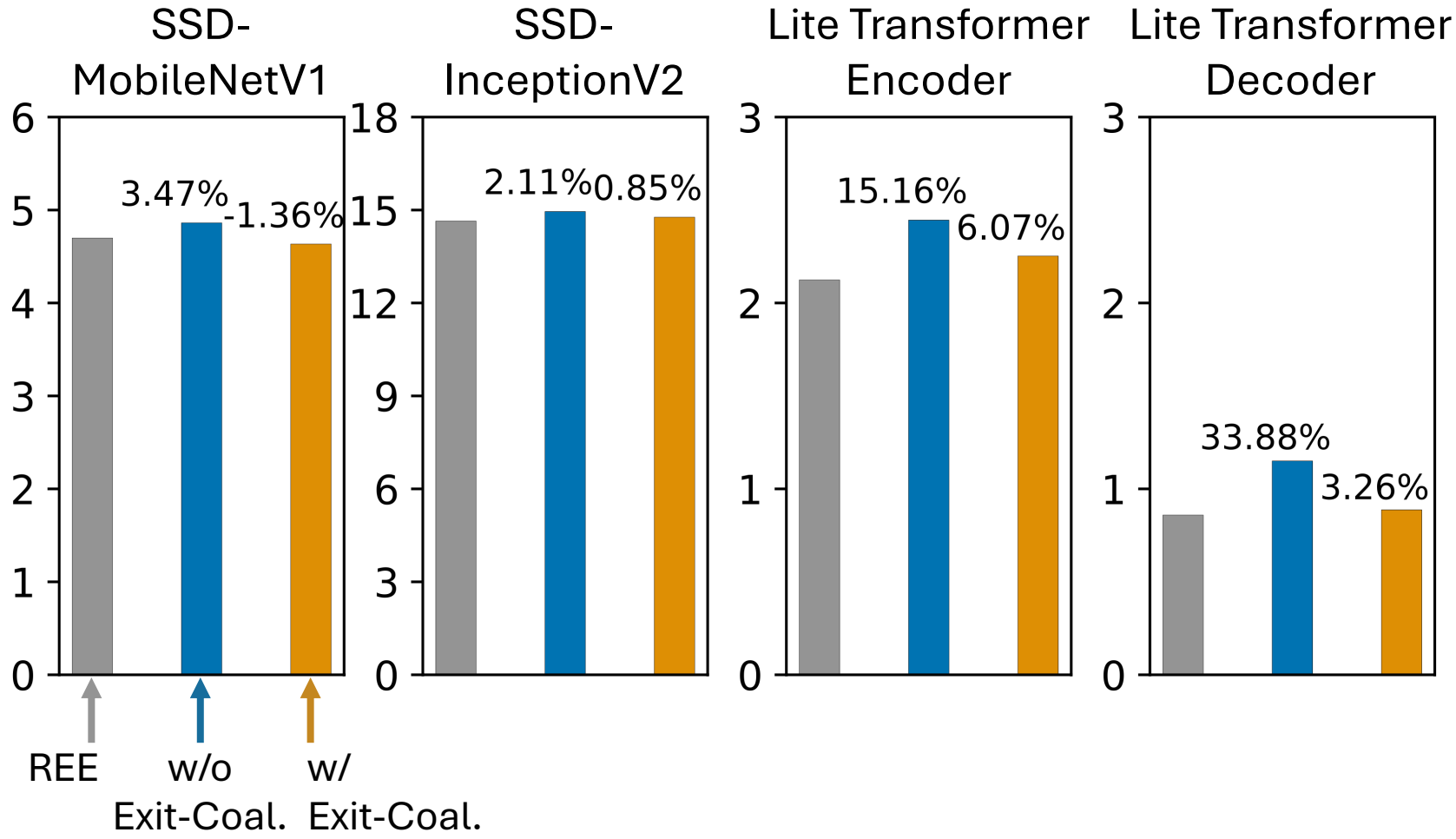
38

Evaluation #2: Exit-Coalescing DNN Execution Planning



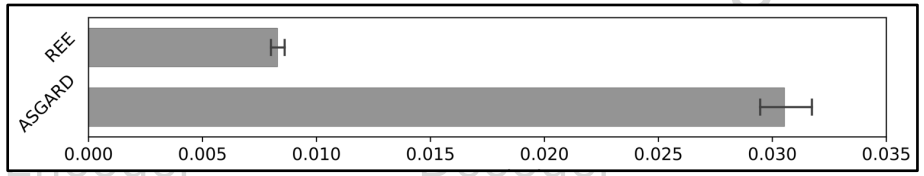
Number of exits: **13** **18** **26** **38**

Evaluation #2: Exit-Coalescing DNN Execution Planning



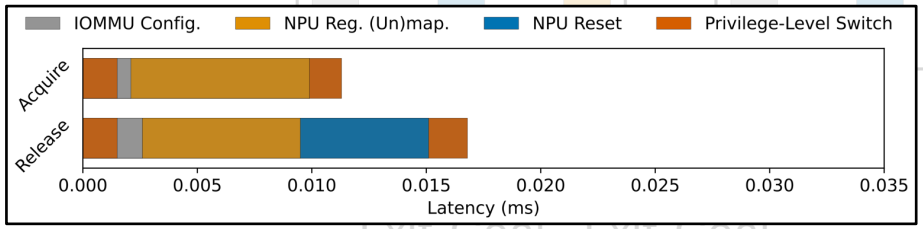
Number of exits: **13 → 2** **18 → 10** **26 → 16** **38 → 20**

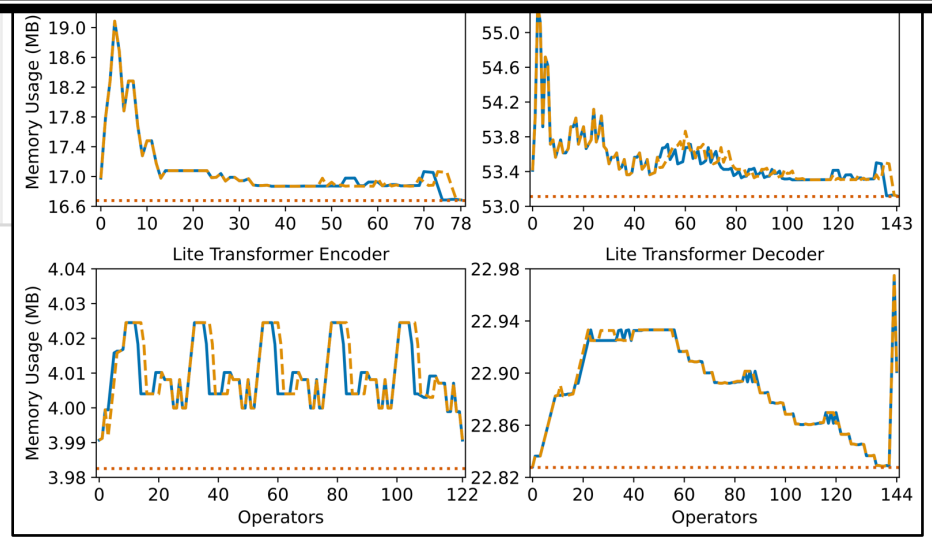
Mode	Component	Size		
		Before	After	(Reduction)
User	Core Utilities	0.831	**0.000**	**(–100.0%)**
	DNN Application	0.023	0.023	(–0.0%)
	DNN Runtime & User-Mode NPU Driver	5.610	5.610	(–0.0%)
	C/C++ Standard Libraries & Linker	4.015	**3.870**	**(–3.6%)**
	Total	10.479	**9.503**	**(–9.3%)**
Kernel	Security Features	7.000	**3.928**	**(–43.9%)**
	Application-Specific Features	6.564	**0.039**	**(–99.4%)**
	Kernel-Mode NPU Driver	0.284	0.284	(–0.0%)
	Core Kernel	3.685	3.685	(–0.0%)
	Total	17.533	**7.936**	**(–54.7%)**



More evaluation results are available in the paper!

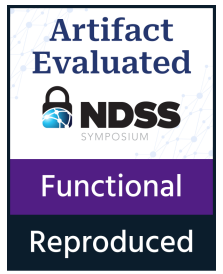
(E.g., security analysis, TCB size, IRQ delivery latency, NPU reassignment latency, memory usage, etc.)





Conclusion

- The **first system** that protects on-device DNNs with virtualization-based TEEs in legacy SoCs.
- The virtualization overheads could be contained with our **system & application-level** optimizations.
- Check our paper for many more details!



Thank you!

Myungsuk Moon

myungsuk@yonsei.ac.kr

Artifact: https://github.com/yonsei-sslab/asgard