presented by

UEFI updates, Secure firmware and Secure Services on Arm

Spring 2018 UEFI Seminar and Plugfest
March 26-30, 2018
Presented by Dong Wei & Matteo Carlini (Arm)

Agenda

- UEFI and SBBR/EBBR Updates
- Secure Services on Arm
- Trusted Firmware-A Updates
- EDK2 Updates

UEFI and SBBR/EBBR Updates

Server Architecture

arm ServerReady

Base System Architecture (BSA)

- Defines hardware requirements

Base Boot Requirements (BBR)

- Defines firmware requirements

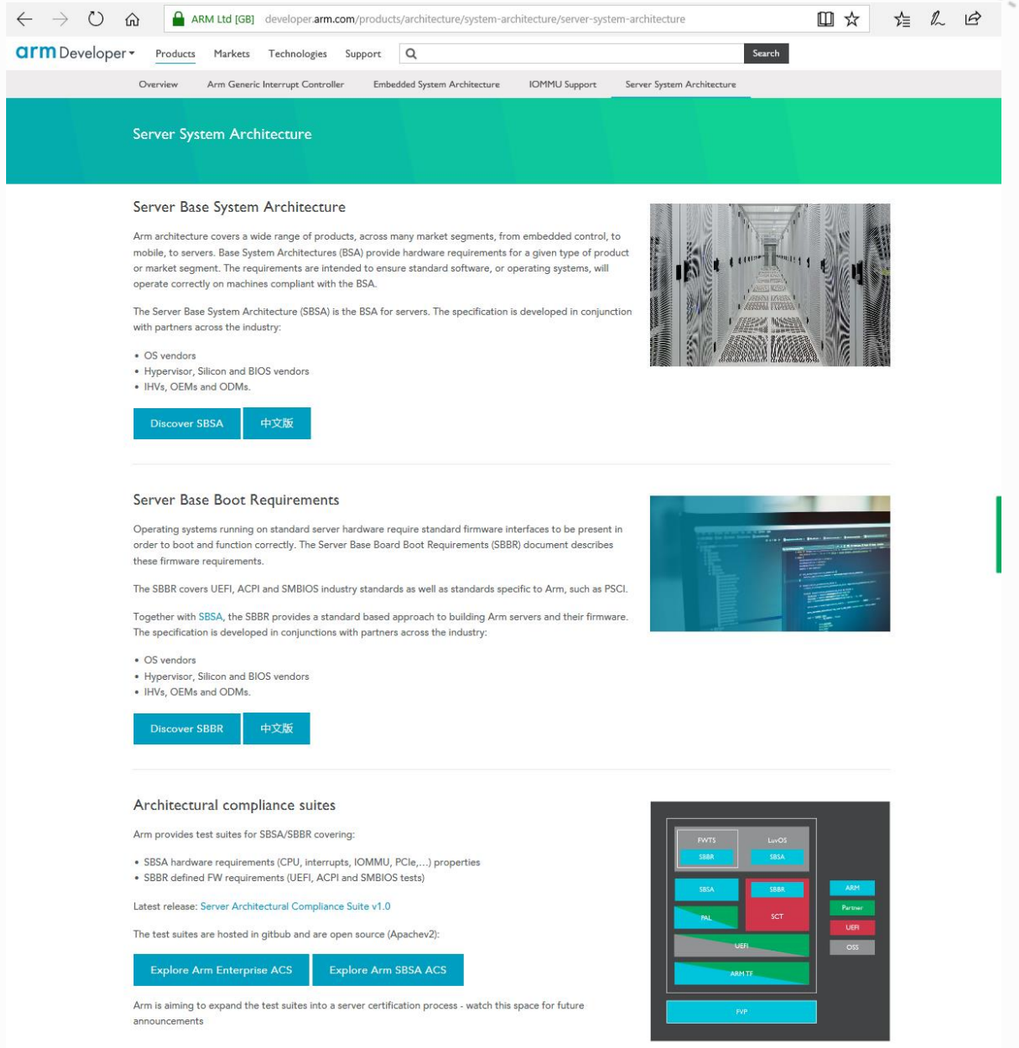
These specifications require a minimum set of hardware and firmware implementations that will ensure OS and firmware will interoperate

SBSA/SBBR are the BSA/BBR for the server systems

- Developed using feedback from vendors across the industry (Silicon vendors, OSVs, Hypervisor vendors, BIOS vendors, OEMs and ODMs)
- SBBR defines the required, recommended and optional UEFI, ACPI and SMBIOS interfaces

SBSA are SBBR are now available at https://developer.arm.com/

- Current versions are SBSA v3.1 and SBBR v1.0. No click through license required.
- SBSA v5.0 and SBBR v1.1 will be available soon



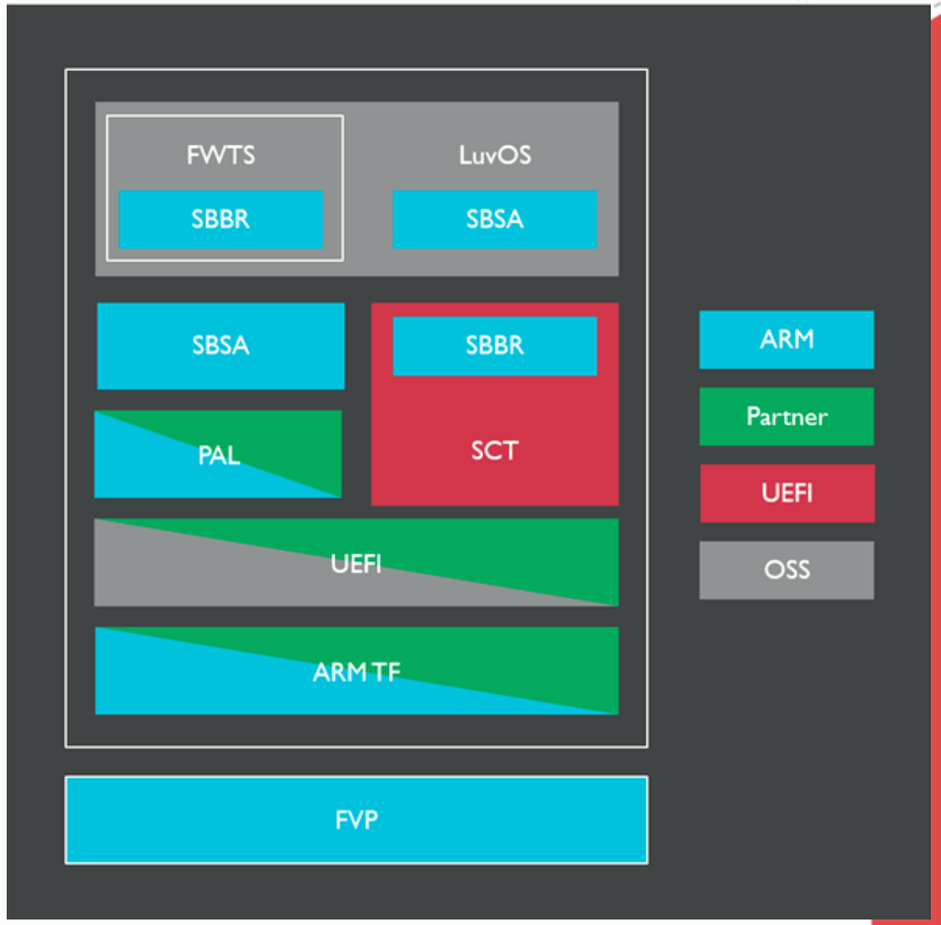
Architectural Compliance Suites

SBSA test covers
- SBSA CPU properties
- SBSA defined system components
- SBSA rules for PCIe integration
  - Based on the PCIe specification
  - Based on standard OS drivers with no quirks enabled

SBBR test covers
- UEFI testing based on the UEFI SCT
- ACPI testing based on FWTS
- SMBIOS testing

V1.3 released!
- https://github.com/ARM-software/sbsa-acs
- https://github.com/ARM-software/arm-enterprise-acs

All Open Source except UEFI SCT
- UEFI Forum BoD is working on a new model



SBBR v1.1

UEFI:
- UEFI PCI Root Bridge IO Protocol Address Translation clarifications
- UEFI GOP implementation clarifications
- UEFI REST Protocol support
- UEFI Capsule Service clarification
- **Native AArch64 image requirements for UEFI applications and drivers**

ACPI:
- ACPI Interrupt-signaled Events support
- ACPI Generic Event Devices support
- ACPI PCI IO Address Translation clarifications
- IORT implementation guidelines

SMBIOS/Management:
- SMBIOS Processor Information
- SMBIOS structure data requirements clarification
- SMBIOS Redfish Host Interface support
- SPMI recommendation removal

Cleanup:
- Clarifications of SSDT being optional
- Clarifications on UEFI Load File and Load File 2 Protocols
- **Updated referenced specifications to: UEFI 2.7, ACPI 6.2, SMBIOS 3.1.1**
- Secondary core boot standardization with PSCI

Security:
- **Secure and Trusted Boot**
- **Secure Firmware Update**

UEFI Option ROM Availability

Architecturally Arm requires the support of AArch64 native binary UEFI drivers

Arm testing room will be open all week to provide:

- Real HW Setup in which to test native AArch64 drivers
- Help and suggestions on how to get your driver recompiled for AArch64

Arm is creating a “getting started guide” and a list of off-the-shelf systems that can be used for continuous testing

Arm is collecting a list of vendors/cards with AArch64 drivers:

- Contact us (uefi@arm.com) if you would like to be on it or if you’d like more information

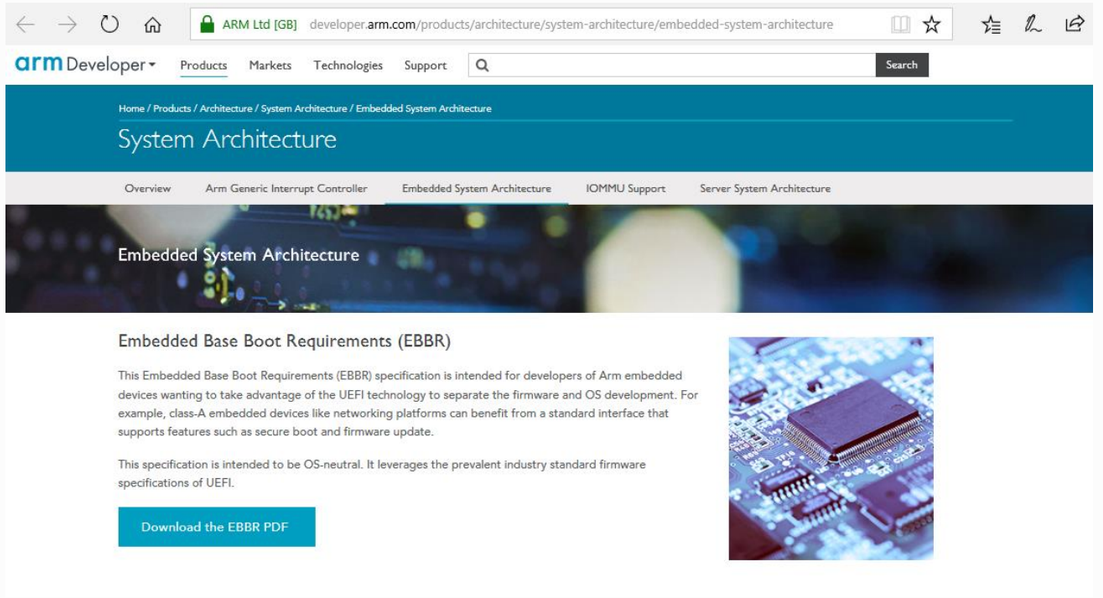
Embedded Architecture

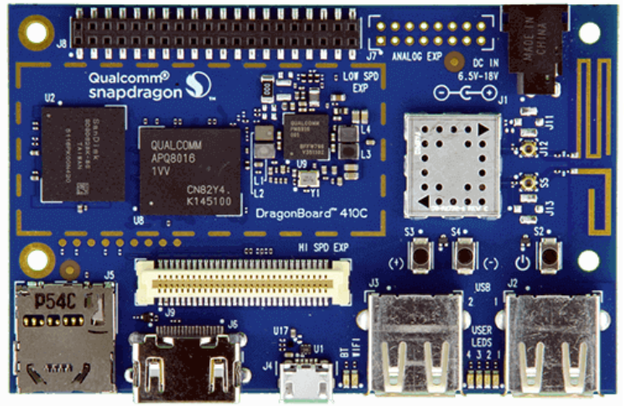
Base System Architecture (BSA)
- Defines hardware requirements

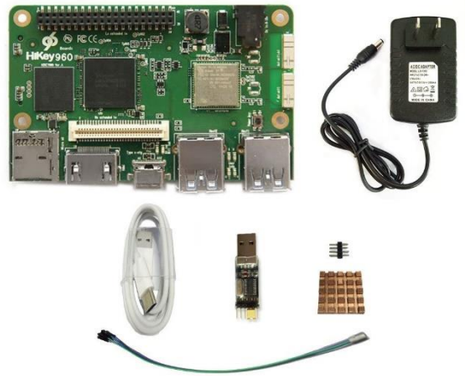
Base Boot Requirements (BBR)
- Defines firmware requirements

These specifications require a minimum set of hardware and firmware implementations that will ensure OS and firmware will interoperate

EBBR is the BBR for the embedded systems
- Under development
- Need review feedback







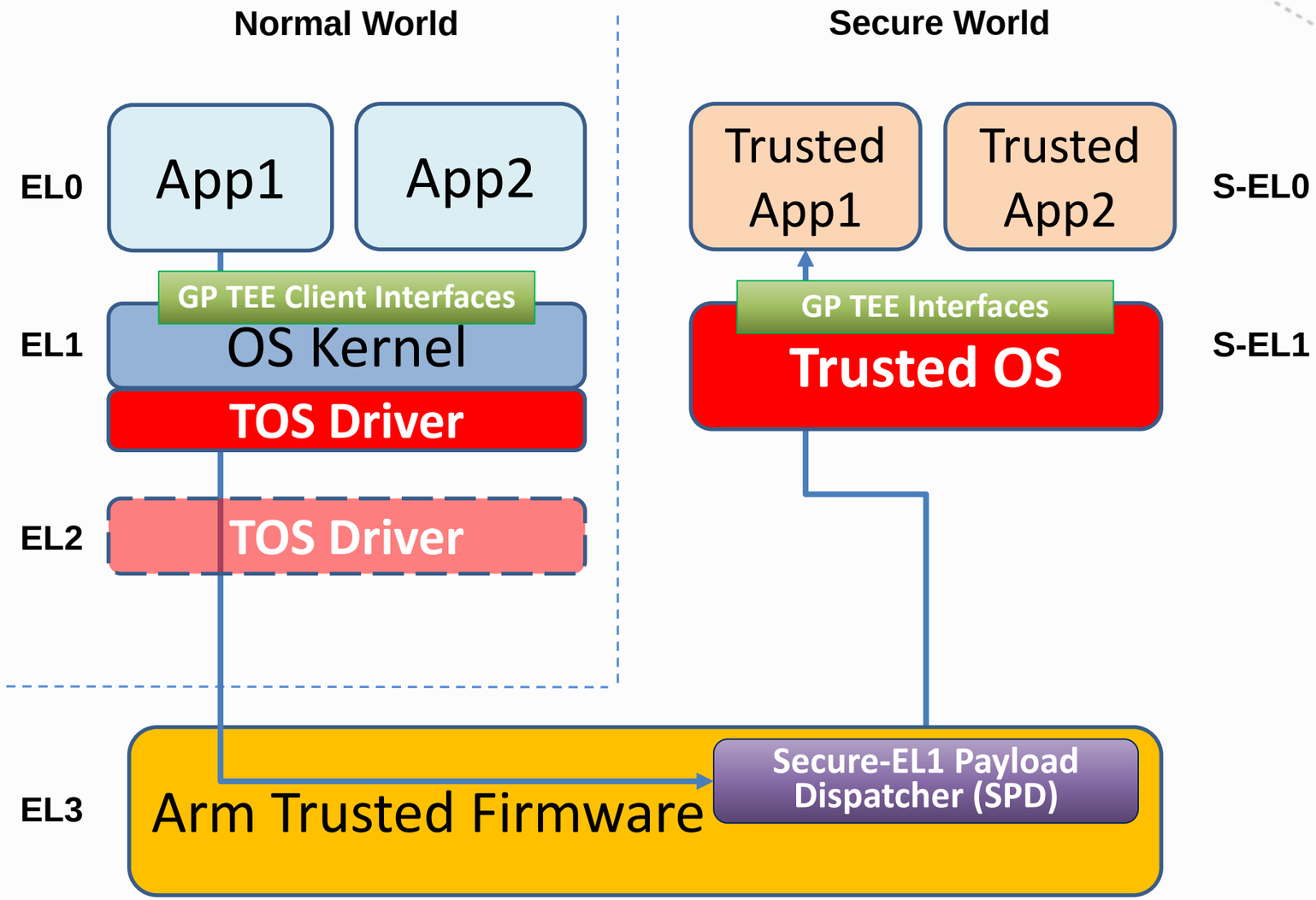
Secure Services on Arm

Secure Services on Arm – Mobile

A pretty stable situation lead by TEE/TOS vendors

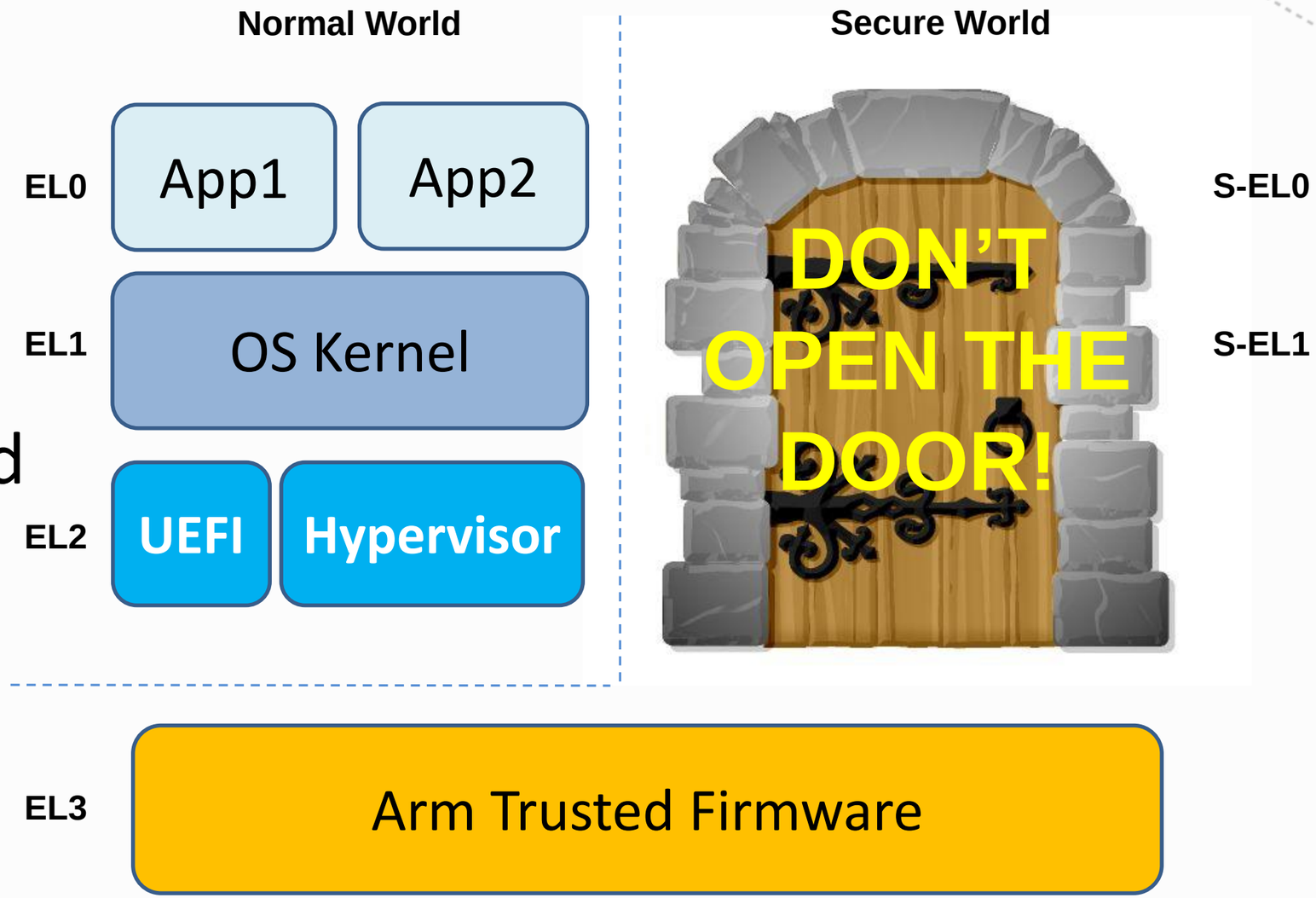
BUT:

- Mainly proprietary code
- No standardised interfaces (but GP TEE ones)
- 1 TEE → 1 TOS constraint

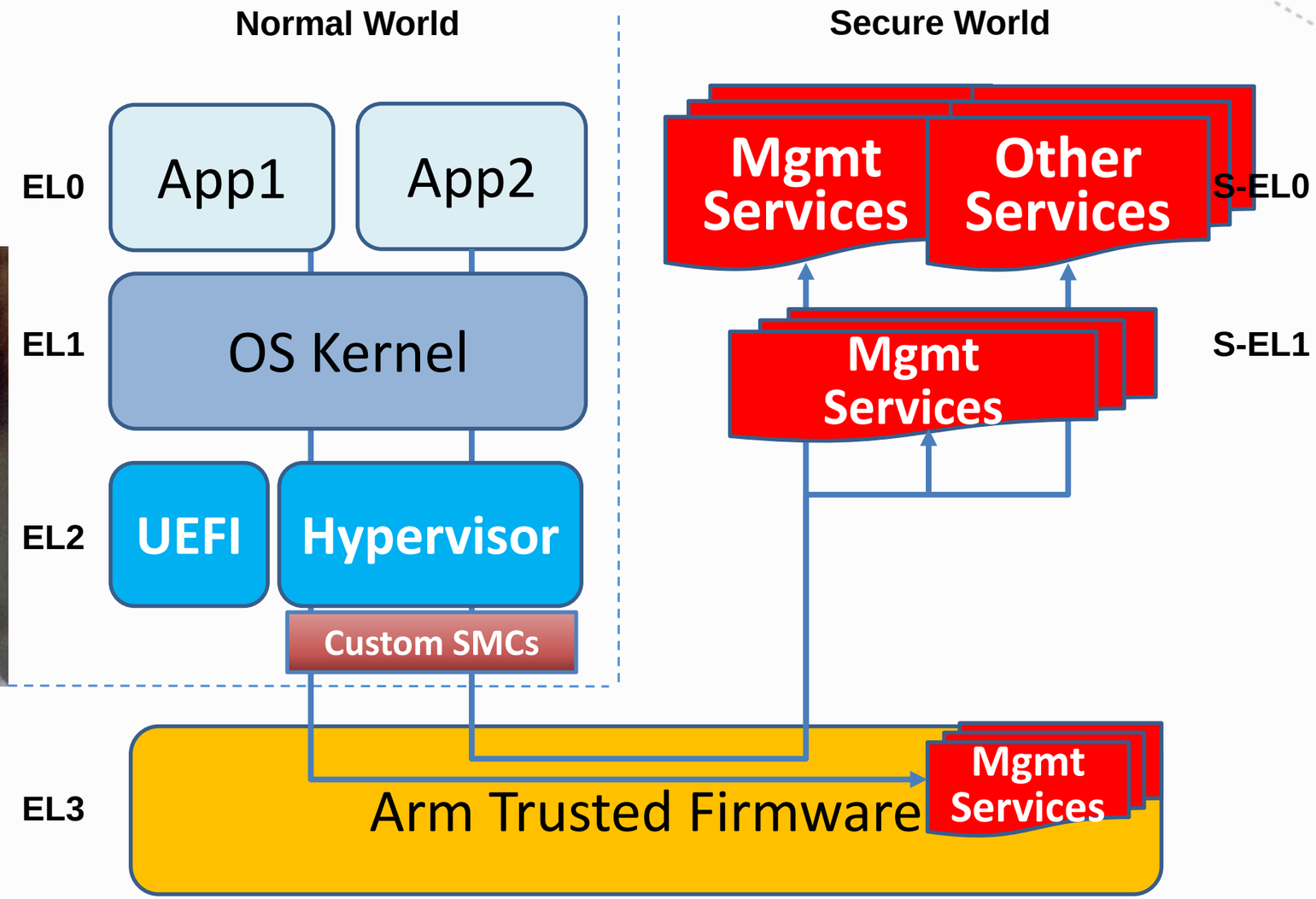


Secure Services on Arm – UEFI

What's really happening instead in the Arm Secure World outside of the Mobile/Trusted OS space??

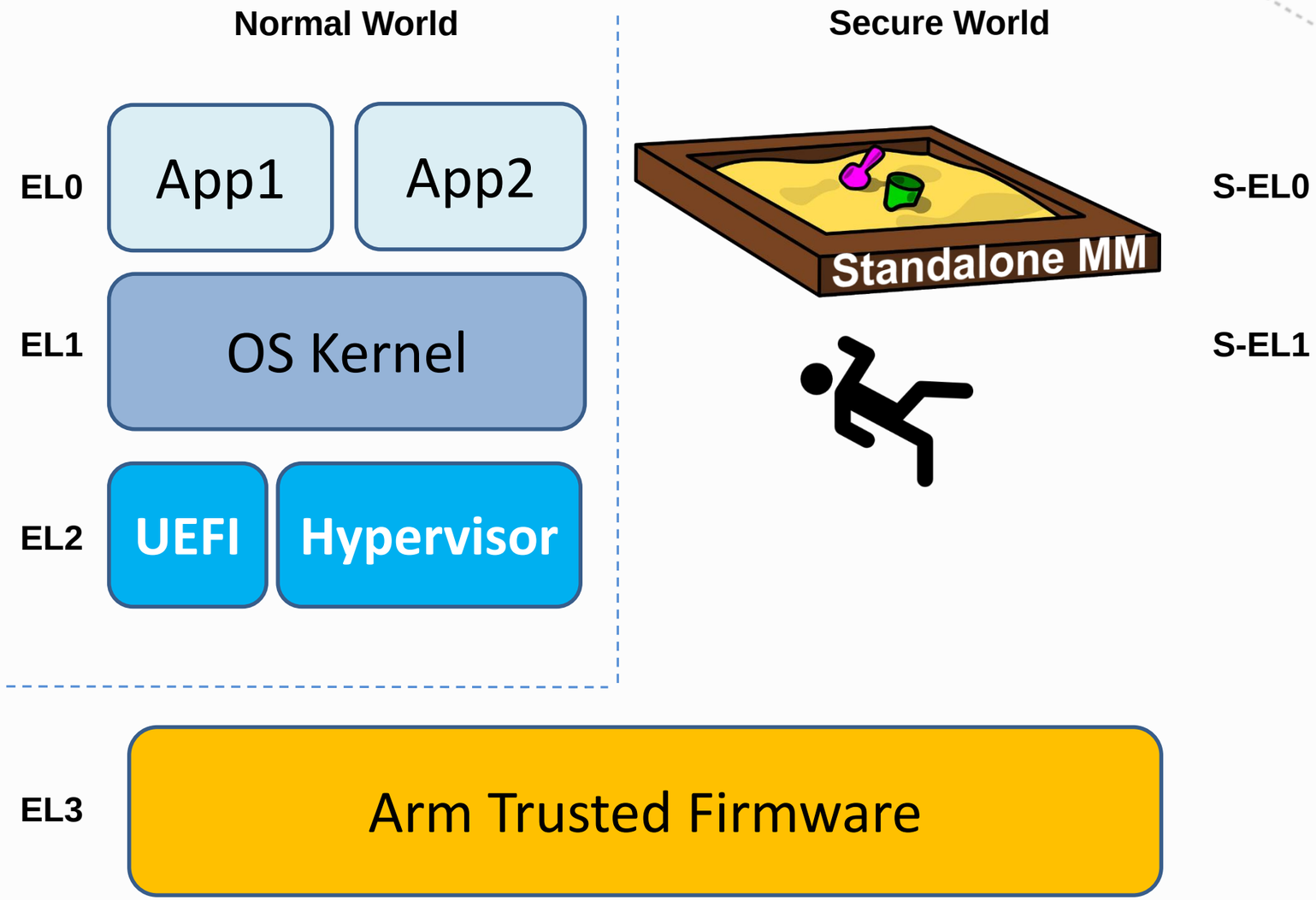


Secure Services on Arm – UEFI



Introducing Secure Partitions

A Secure Partition is an unprivileged software sandbox environment running in the Secure World, under the control of privileged software, to instantiate PI Standalone Management Mode, in order to execute MM (secure) services.



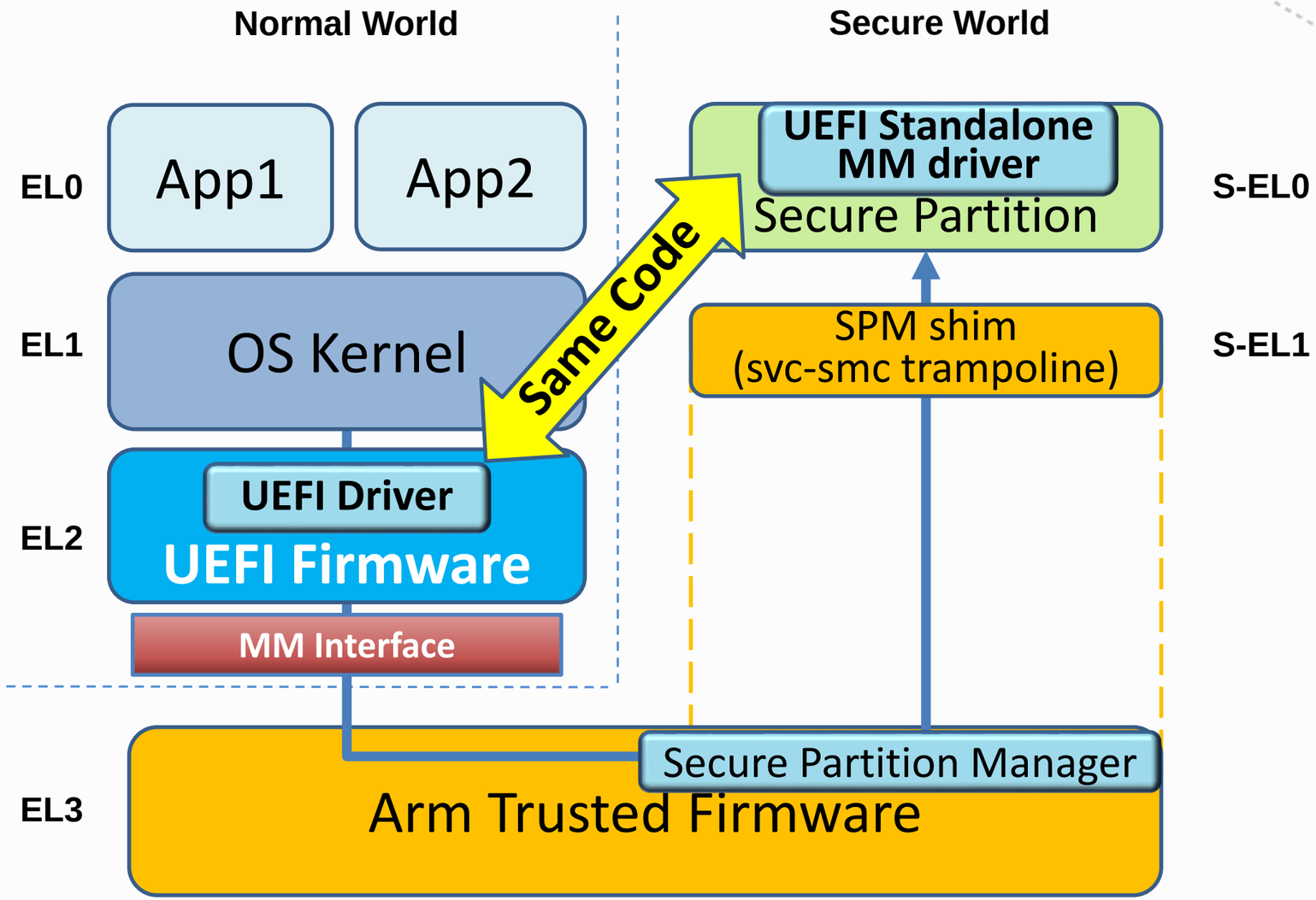
Secure Partitions – Use Cases

1. Secure persistent Storage
   - Secure Variable access
   - Firmware Update
2. Management Services
   - Errata handling
   - BMC communication
   - RAS Error Handling
3. RNG
4. Others?

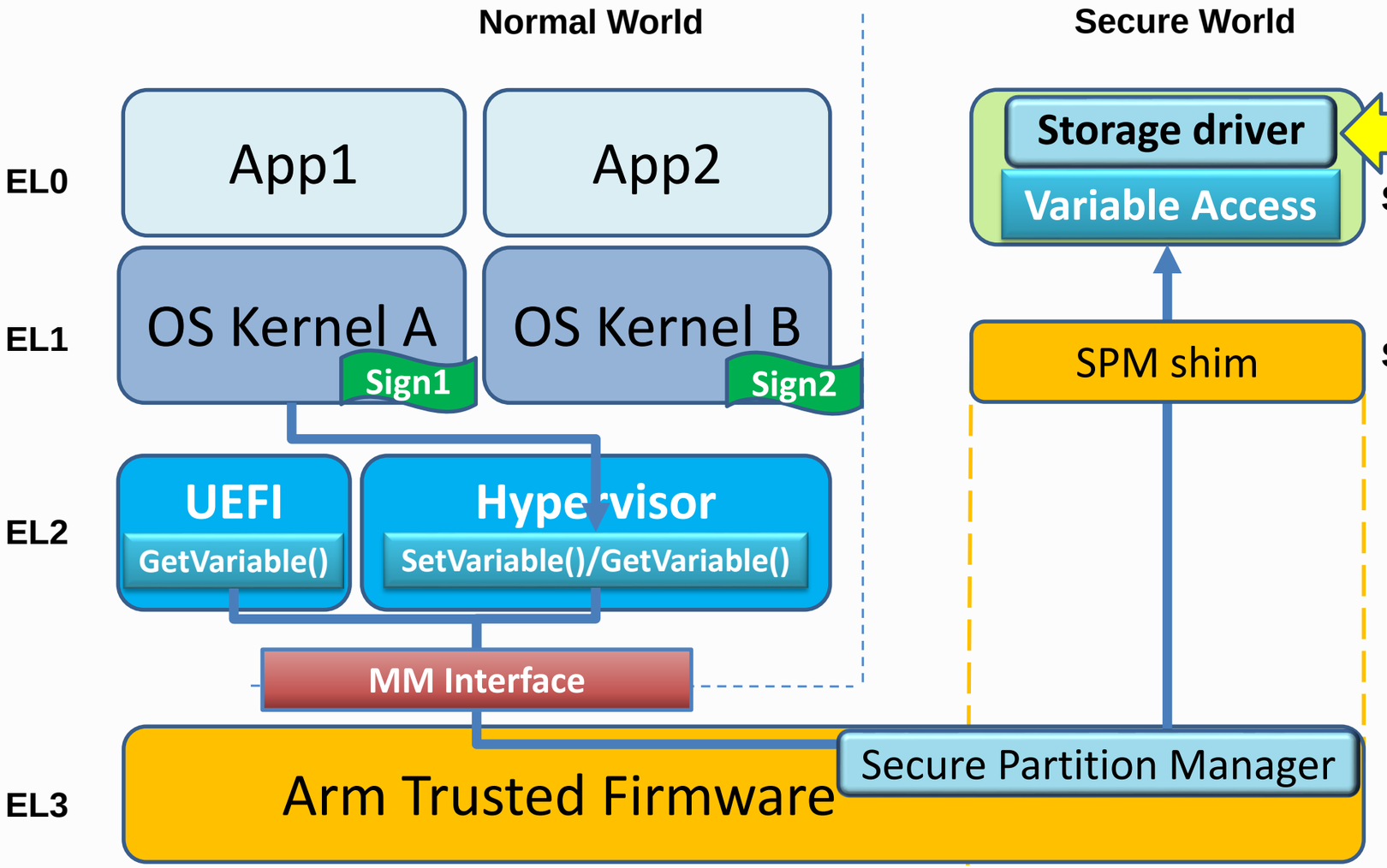
Software Architecture

Main design goals:

- Isolated execution context
- Limited access to system resources
- OS agnostic
- Leverage Arm MM Interface Specification, Arm TrustZone & UEFI Standalone MM
- Well defined interfaces
- Code reuse between normal/secure world whenever possible
- Reduced services code into privileged firmware (EL3)



Real Case: Secure Variable access

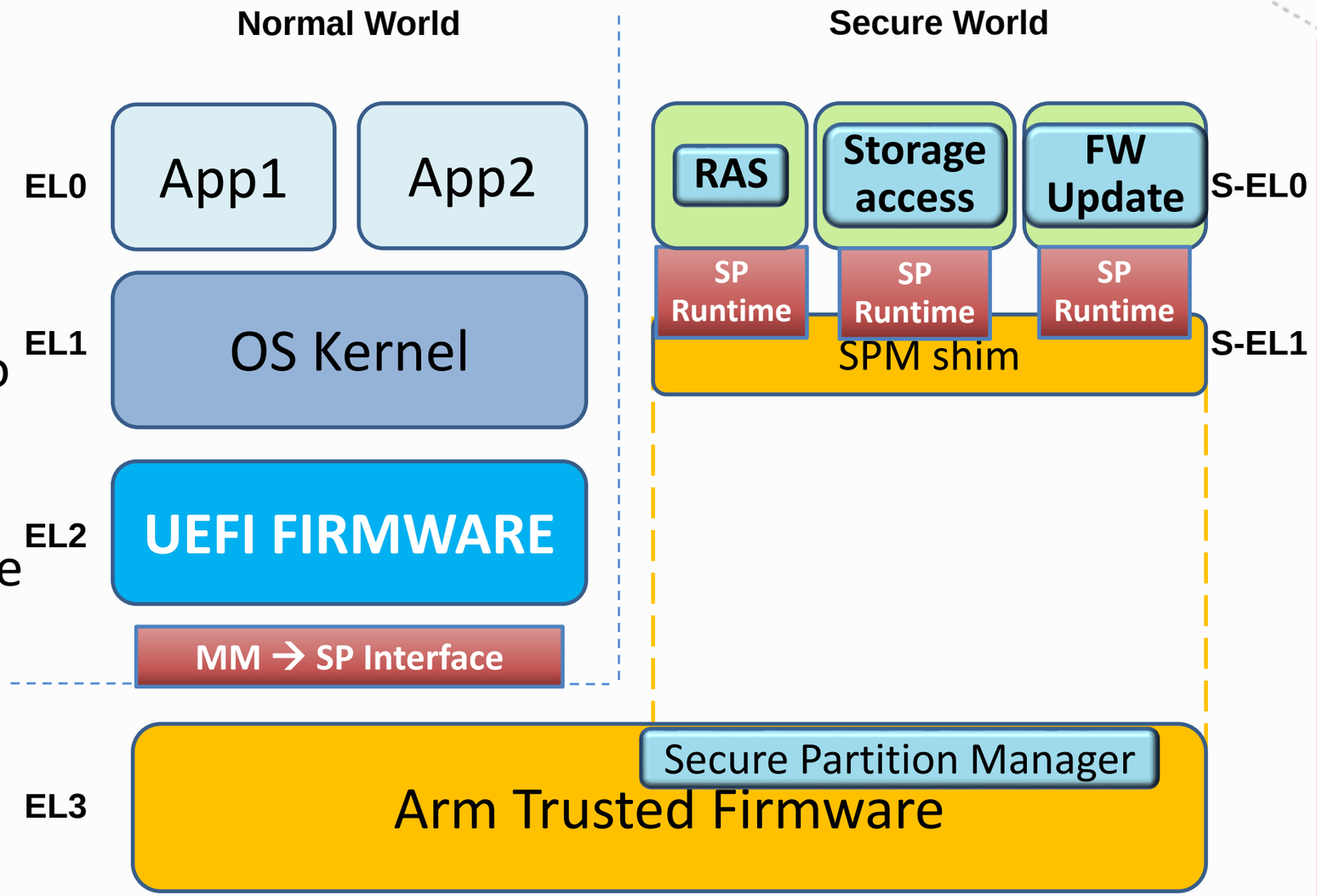


UEFI Variables in Secure Storage:

- **Keys**
- **SignatureDB**

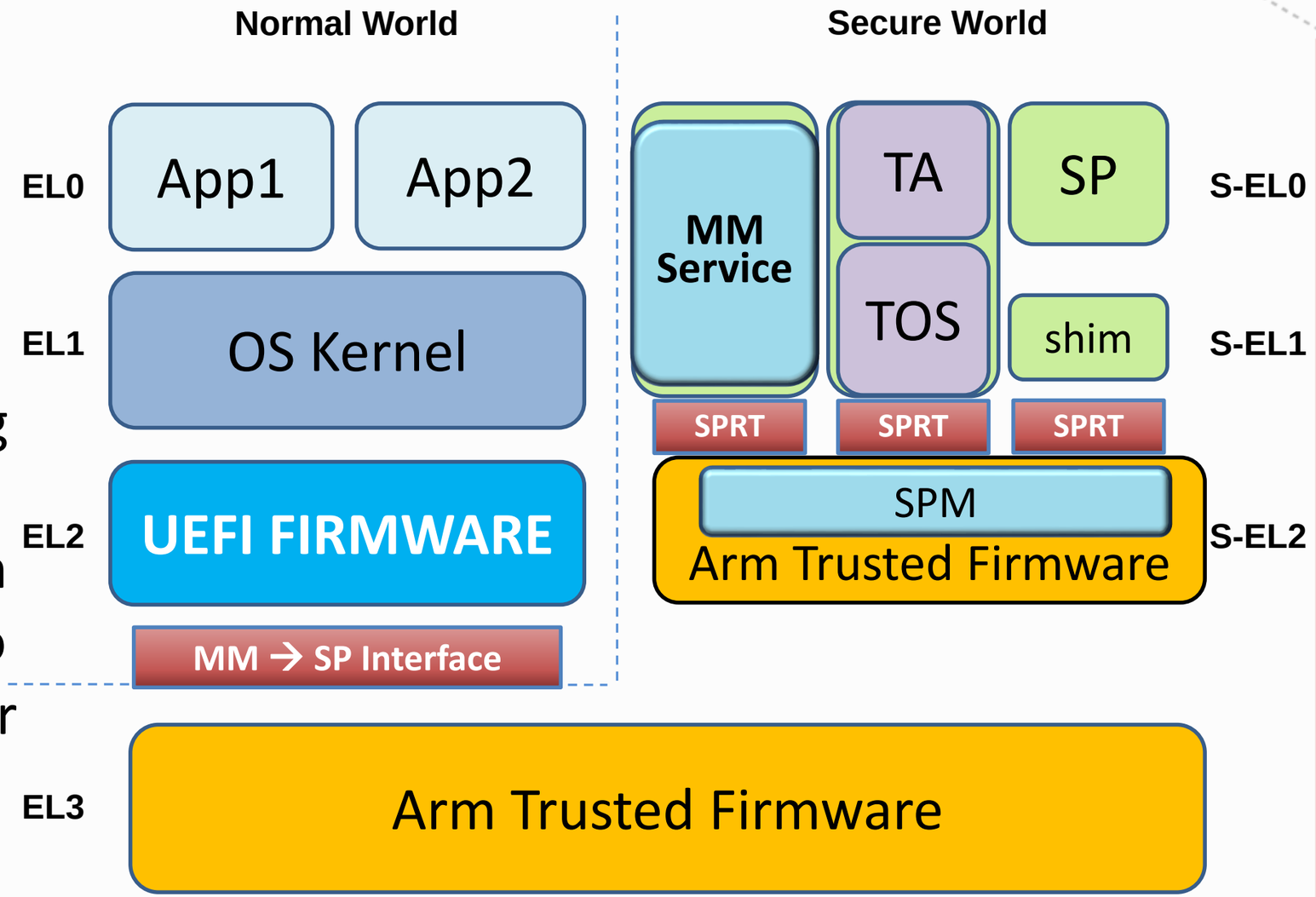
Evolutions (1) – Multiple Services

- Multiple parallel Secure Partitions enabling concurrent Secure Services to run at the same time at S-EL0
- Each Secure Service into each SP is isolated from any other
- MM Interface will evolve into **Secure Partition Client Interface (SPCI)**
- **SP Runtime interface (SPRT)** in the Secure World



Evolutions (2) – Secure-EL2

Future Arm architectures (v8.4 onwards) will introduce a Secure-EL2 exception level that will enable scenarios with multiple TEE/TOS running in parallel as well as allowing coexistence with MM services running into Secure Partitions at either S-EL0 or S-EL1



Trusted Firmware-A Updates

Trusted Firmware-A Updates

Secure Partition Manager (SPM) responsibilities:

- Allocate resources requested by Secure Partitions
- Perform architectural and system setup required by the Secure Partition to fulfil a service request
- Implement standard interfaces (defined by current and upcoming specifications)
  - For initialising a Secure Partition
  - Used by a Secure Partition to fulfil service requests
  - Used by the Non-secure world for accessing the services exported by a Secure Partition (MM Interface)

SPM vs SPD:

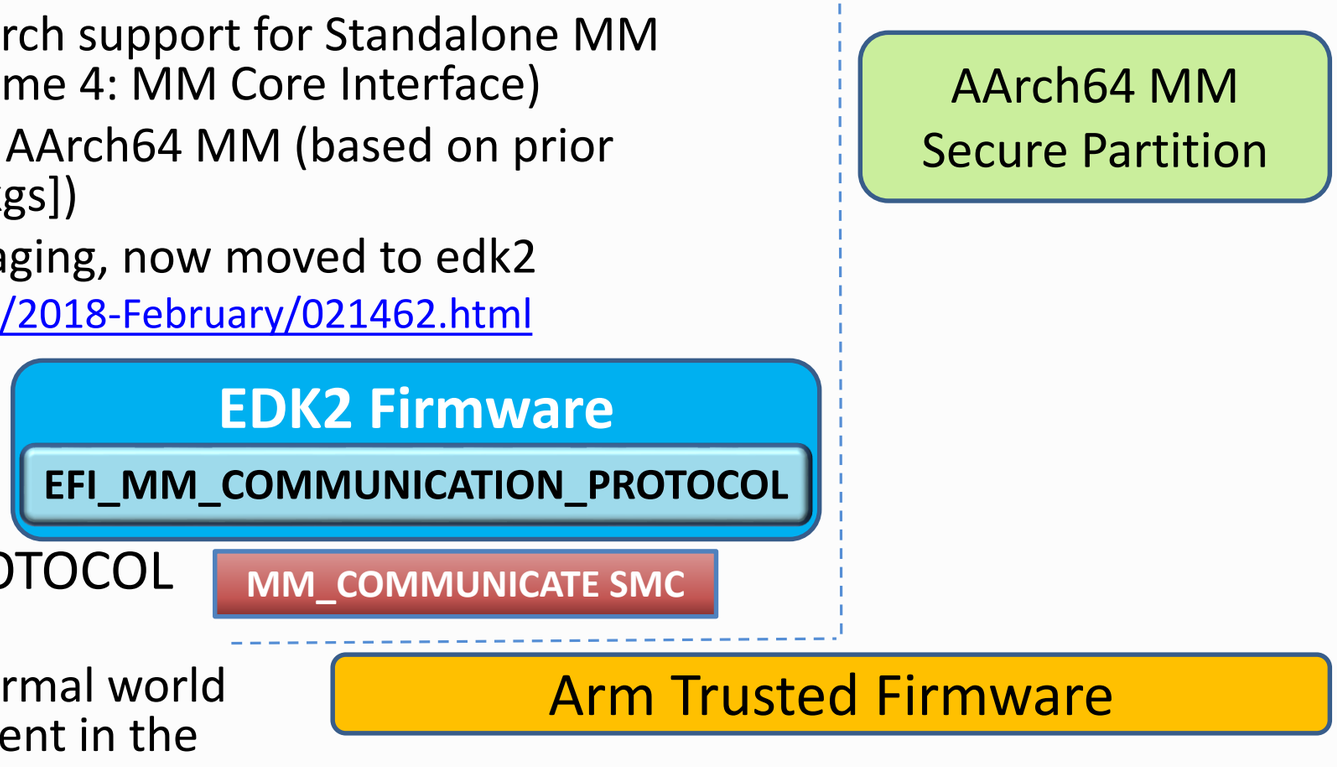
- SPM and SPD are **mutually exclusive**
- SPD does NOT handle S-EL0 TAs: all management handed over to Trusted OS at S-EL1
- SPM instead takes directly care of all lifecycle of SPs (at EL3 today, potentially at S-EL2 in future evolutions)
- SPM will track the evolutions of the MM/SPCI/SPRT Arm Specifications

EDK2 Updates

EDK2 – StandaloneMmPkg & MM

- StandaloneMmPkg
  - New package for hosting multi-arch support for Standalone MM (as per PI Specification v1.5 Volume 4: MM Core Interface)
  - Newly implemented support for AArch64 MM (based on prior work on x86 platforms [Smm*Pkgs])
  - Initially developed under edk-staging, now moved to edk2

https://lists.01.org/pipermail/edk2-devel/2018-February/021462.html



- EFI_MM_COMMUNICATION_PROTOCOL
  - AArch64 DXE runtime driver for communication between the Normal world firmware and the MM environment in the Secure world

https://lists.01.org/pipermail/edk2-devel/2018-January/020163.html

  - Leverage the MM_COMMUNICATE SMC defined in the Arm MM Interface Specification

References

- Arm MM Interface Specification
  - http://infocenter.arm.com/help/topic/com.arm.doc.den0060a/DEN0060A_ARM_MM_Interface_Specification.pdf
- UEFI PI Specifications
  - http://www.uefi.org/sites/default/files/resources/PI_Spec_1_6.pdf
- Arm Trusted Firmware-A – Secure Partition Manager design document
  - https://github.com/ARM-software/arm-trusted-firmware/blob/master/docs/secure-partition-manager-design.rst
- Arm Secure EL2 extension
  - https://community.arm.com/processors/b/blog/posts/introducing-2017s-extensions-to-the-arm-architecture
- StandaloneMmPkg
  - https://lists.01.org/pipermail/edk2-devel/2018-February/021462.html
- EFI_MM_COMMUNICATION_PROTOCOL
  - https://lists.01.org/pipermail/edk2-devel/2018-January/020163.html

Questions?

Thanks for attending the Spring 2018 UEFI Plugfest

For more information on the UEFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by

arm