2023-2024学年春季学期

计算机体系结构安全
Computer Architecture Security

授课团队：史岗，陈李维

计算机体系结构安全

Computer Architecture Security

[第7次课] 安全体系结构原理

授课教师：史岗

授课时间：2024.4.8

内容概要

- 安全体系结构现状与不足
- 安全体系结构框架与原理
- 总结

内容概要

- 安全体系结构现状与不足
  - 平台安全架构（Platform Security Architecture)
  - 可信执行环境（Trust Execution Environment)
  - 存在的不足之处
- 安全体系结构框架与原理
- 总结

内容概要

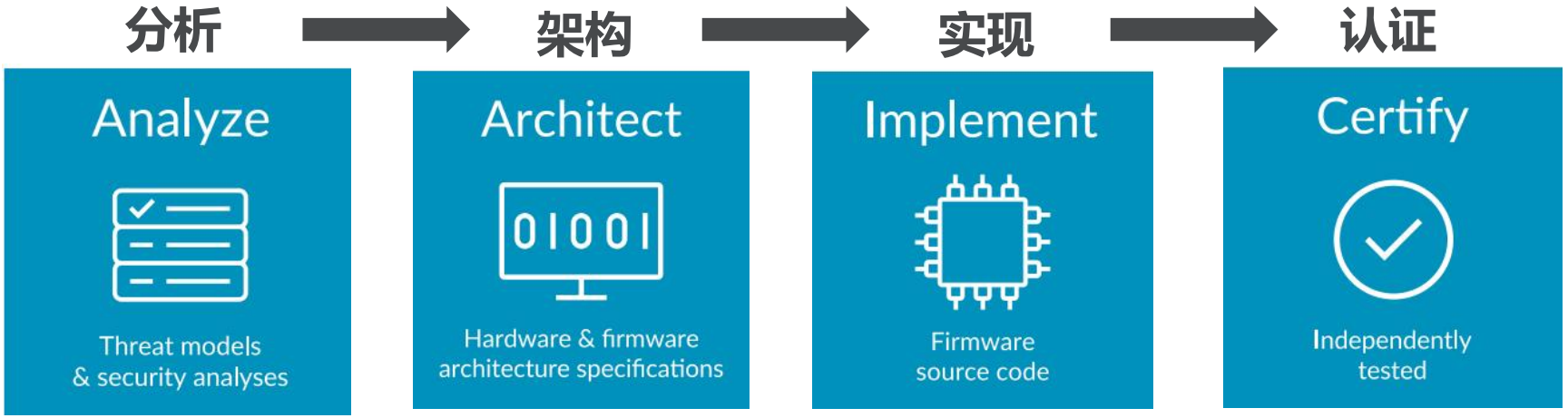
- 安全体系结构现状与不足
  - 平台安全架构（Platform Security Architecture)
  - 可信执行环境（Trust Execution Environment)
  - 存在的不足之处
- 安全体系结构框架与原理
- 总结

平台安全架构（Platform Security Architecture，PSA）

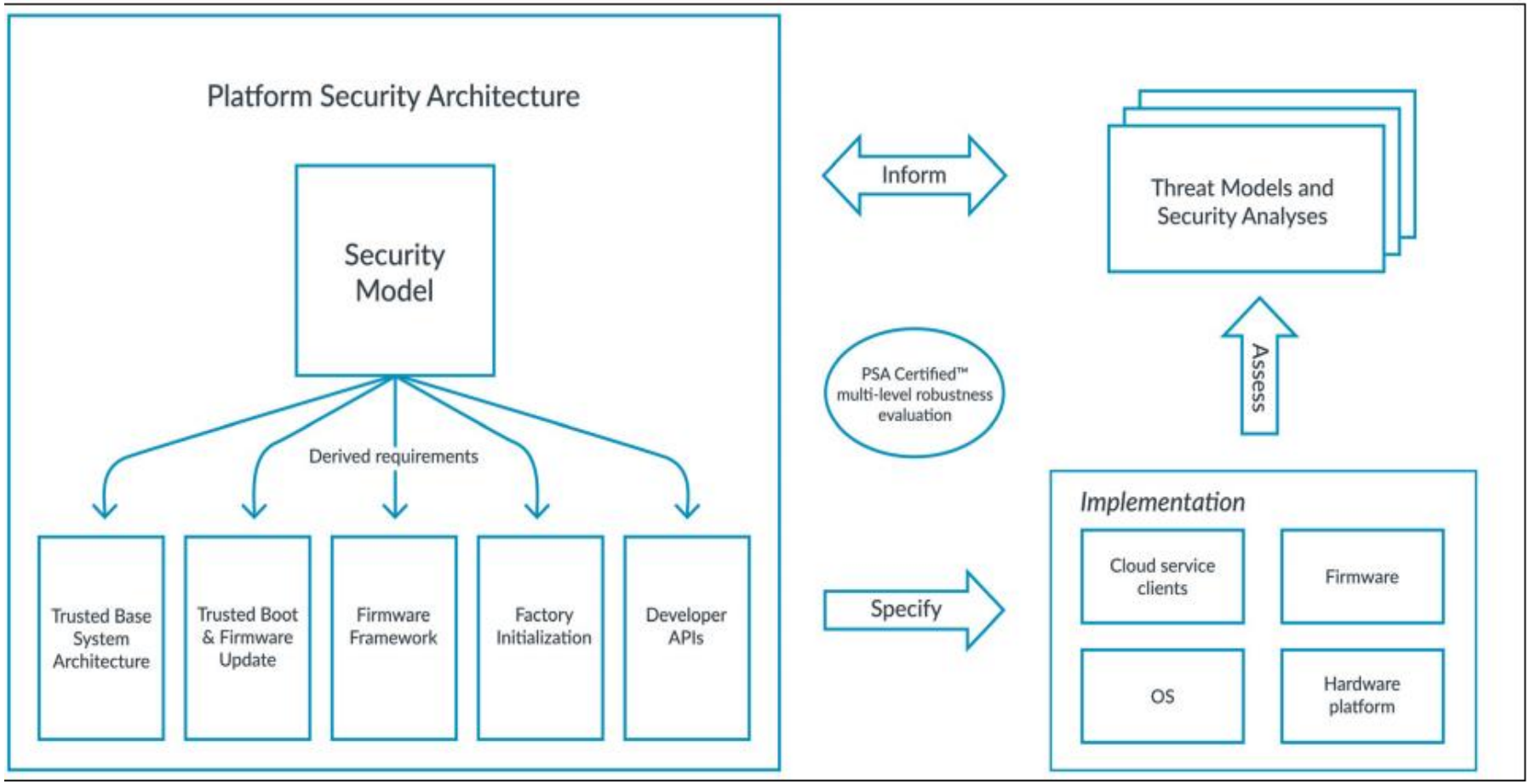
- PSA：

由ARM公司提出的一种从设备设计初始就引入安全要素的框架。包括**威胁模型和安全分析文档、硬件和固件架构规范、API和API测试套件、安全评估和认证**等4个方面内容。

它由四个阶段组成，每个阶段都有相应的文档和指南。



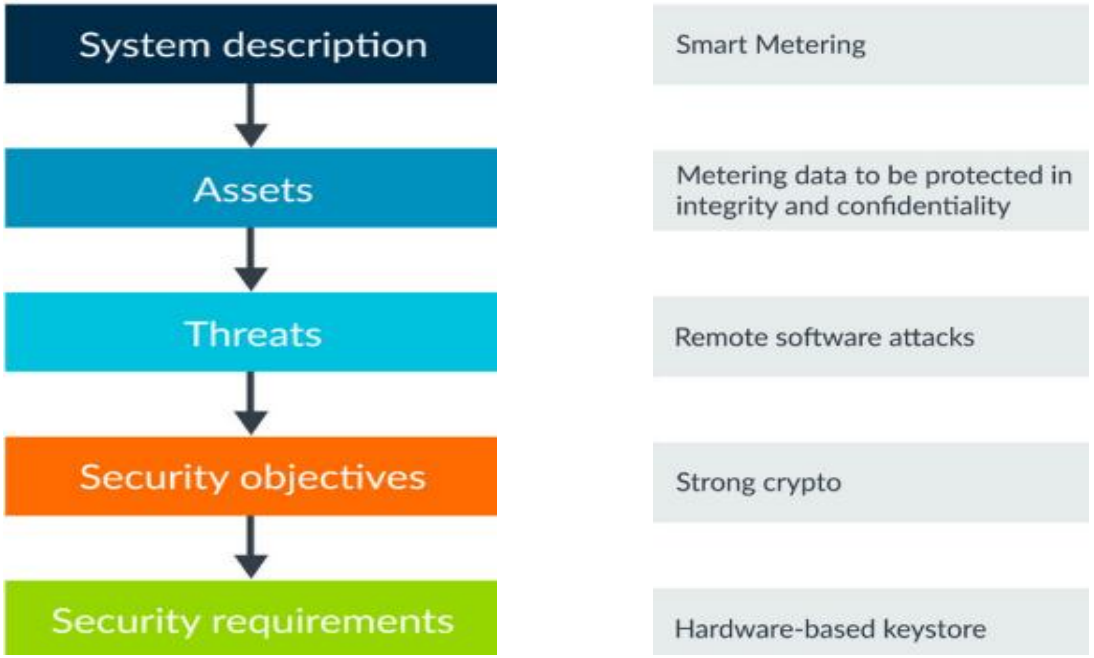
- 四个阶段的关系：


Platform Security Architecture
Security Model
Derived requirements
Trusted Base System Architecture
Trusted Boot & Firmware Update
Firmware Framework
Factory Initialization
Developer APIs
Inform
Threat Models and Security Analyses
Assess
PSA Certified™ multi-level robustness evaluation
Specify
Implementation
Cloud service clients
Firmware
OS
Hardware platform

- 阶段一：建立威胁模型，进行安全分析
  - 评估需要保护的资产
  - 分析所有潜在的威胁
  - 威胁的范围和严重性
  - 攻击者的类型和漏洞利用的方式

安全目标
&
安全功能需求

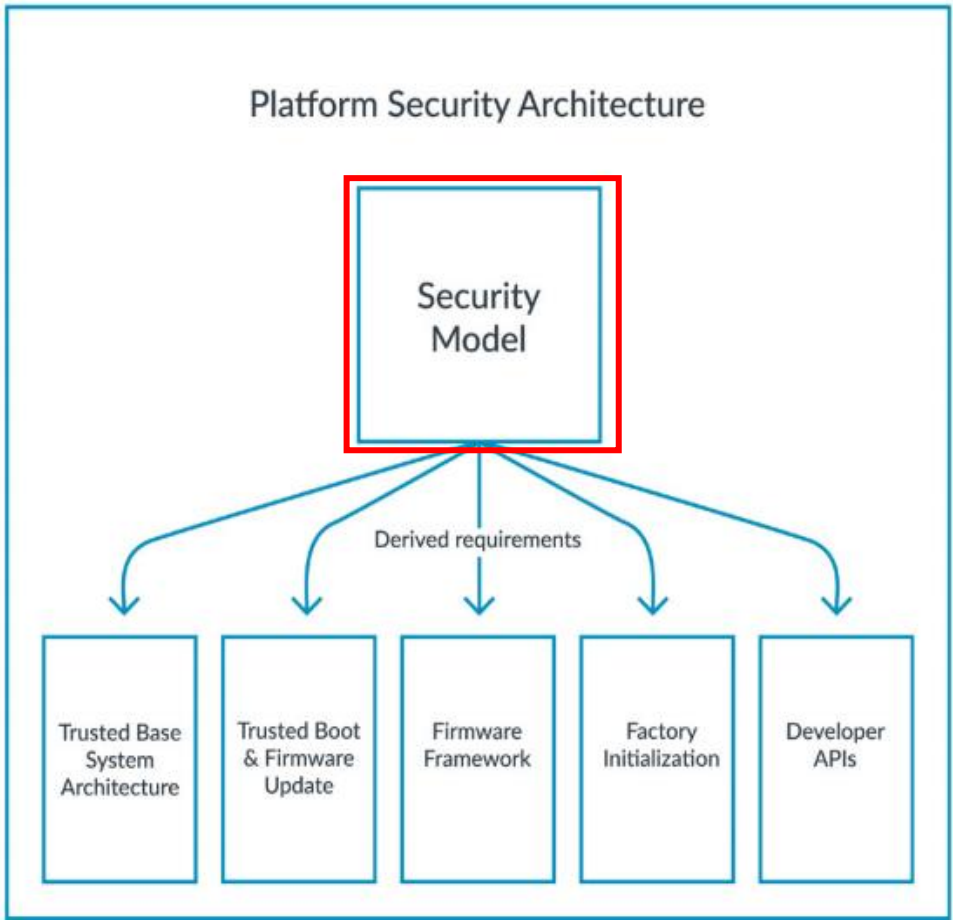
智能电表为例：



阶段二：构建架构规范

安全模型

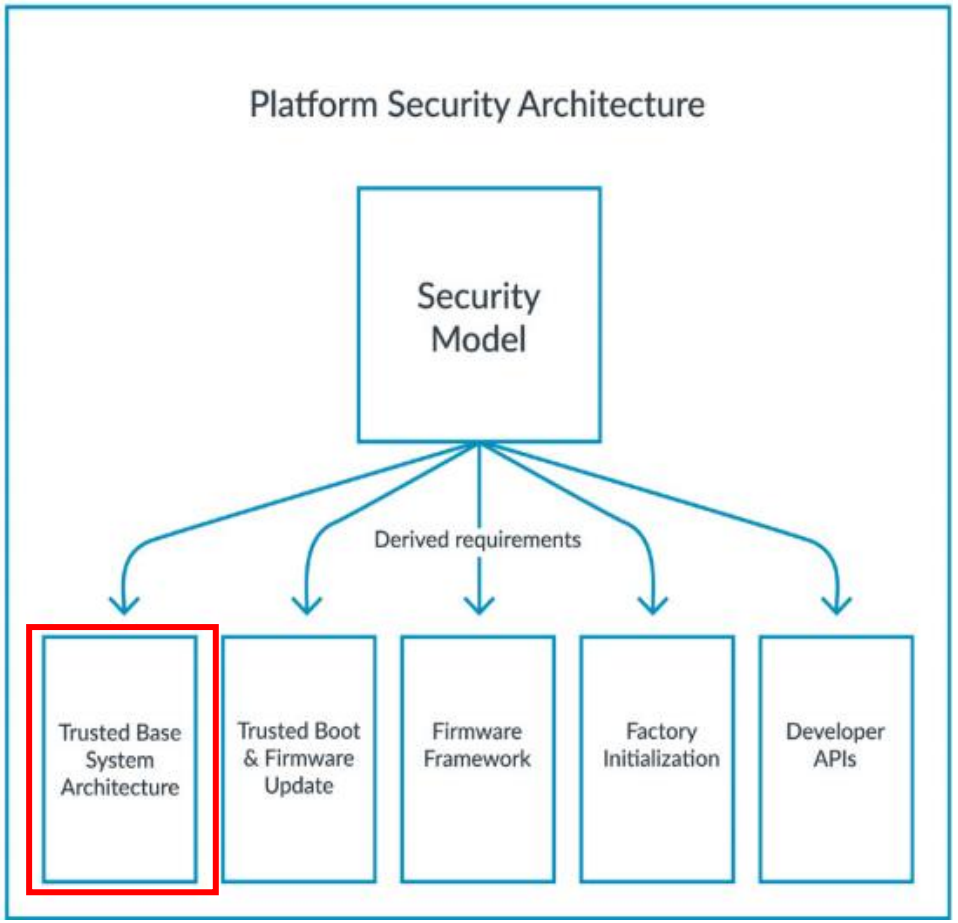
- 定义了整体安全架构
- 由威胁模型和安全需求驱动
- 覆盖三个主要方面
  - Cache一致性互连
  - 通用中断控制器
  - 系统内存管理单元



阶段二：构建架构规范

可信基系统架构

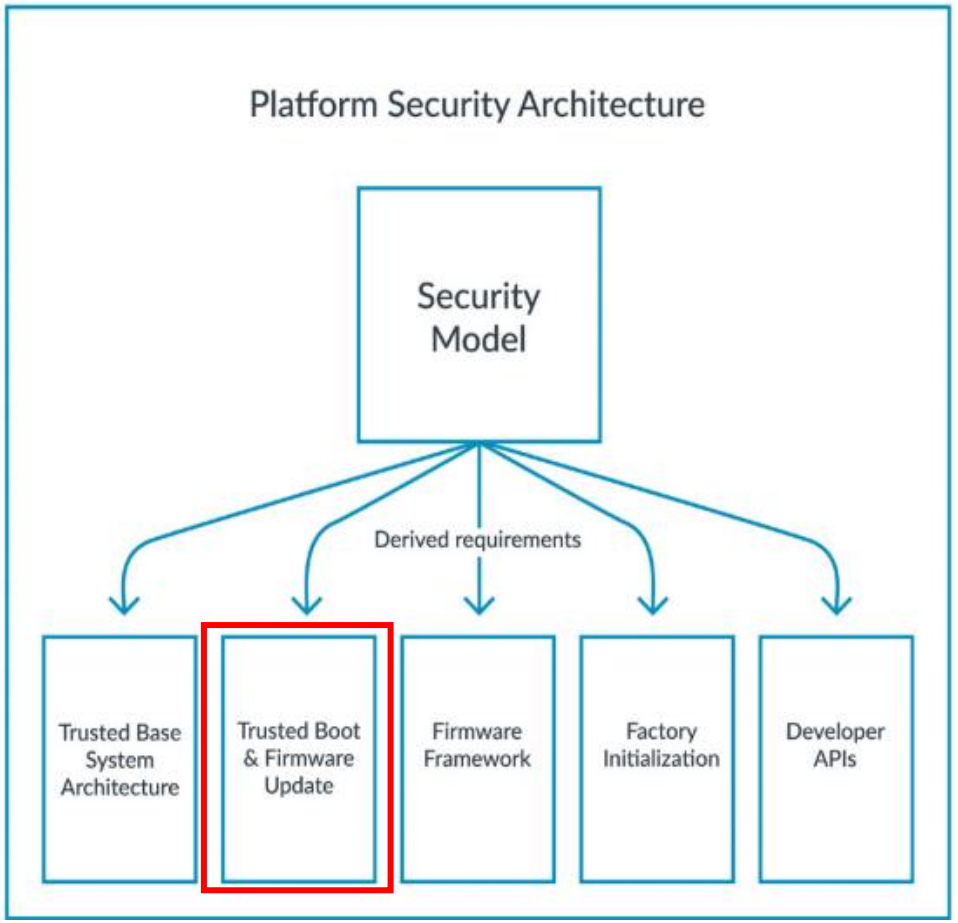
- 是一套 SoC 硬件要求，是基于ARM处理器系统的安全原则
- 是Firmware Framework运行的底层支撑
- 包括：可信根、密钥库、安全固件的更新机制、调试接口要求、加解密加速、可信软件与不可信软件的隔离等



阶段二：构建架构规范

可信启动和固件更新

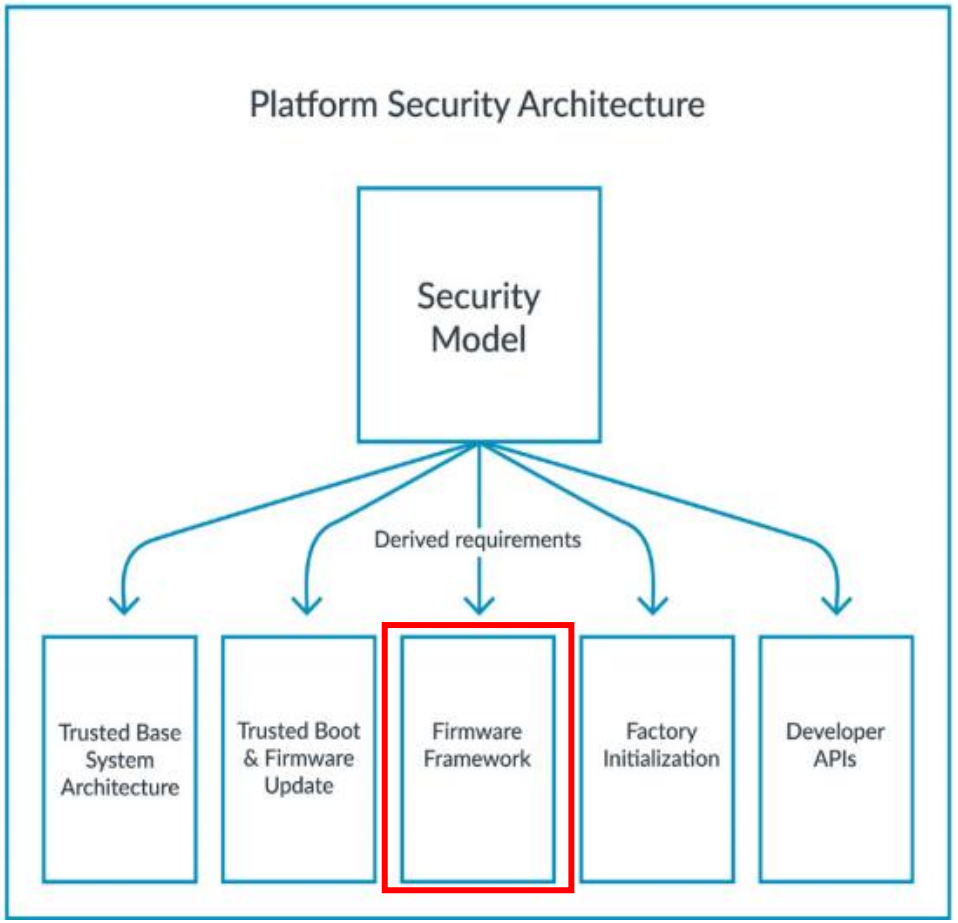
- 对引导过程进行认证以建立安全的运行时服务
- 使用加密和密钥对固件更新进行验证和授权
- 固件更新代理负责更新的安全性



阶段二：构建架构规范

固件框架

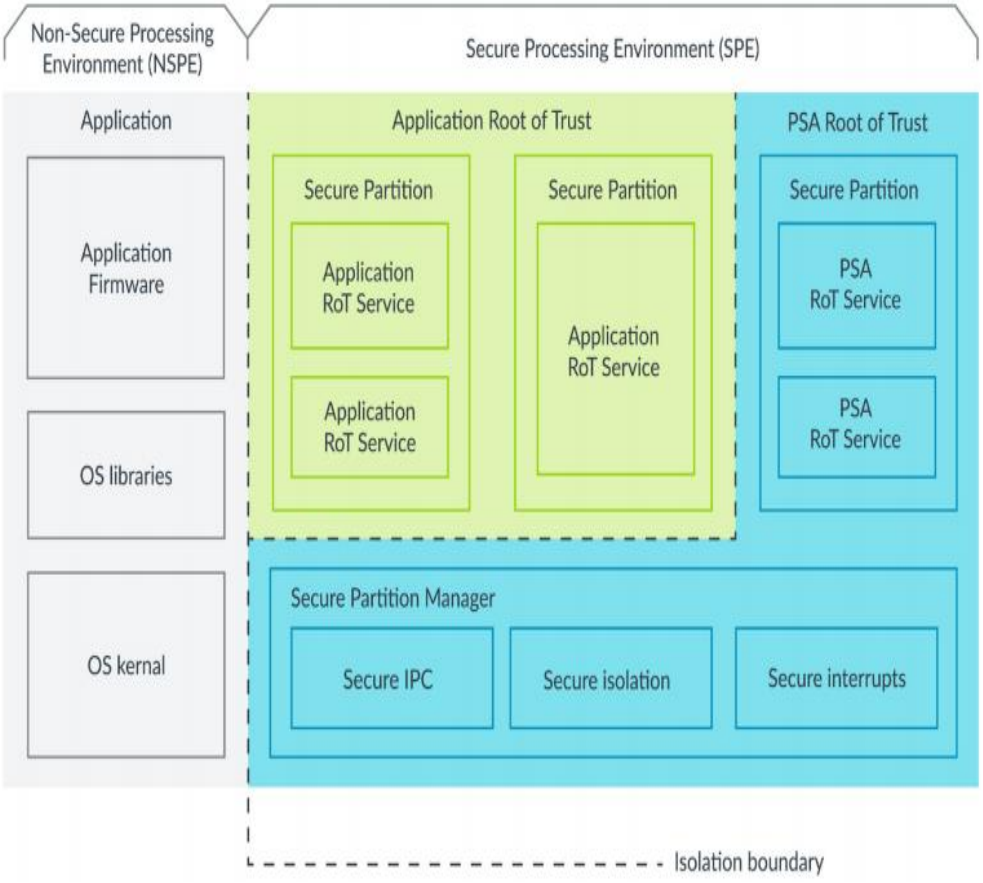
- 定义标准的接口和框架，以隔离可信功能
  - 描述可信和不可信固件的隔离运行时环境（分区）
  - 每个分区中功能和资源
  - 用于分区间进行通信的进程间通信机制



阶段二：构建架构规范

固件框架

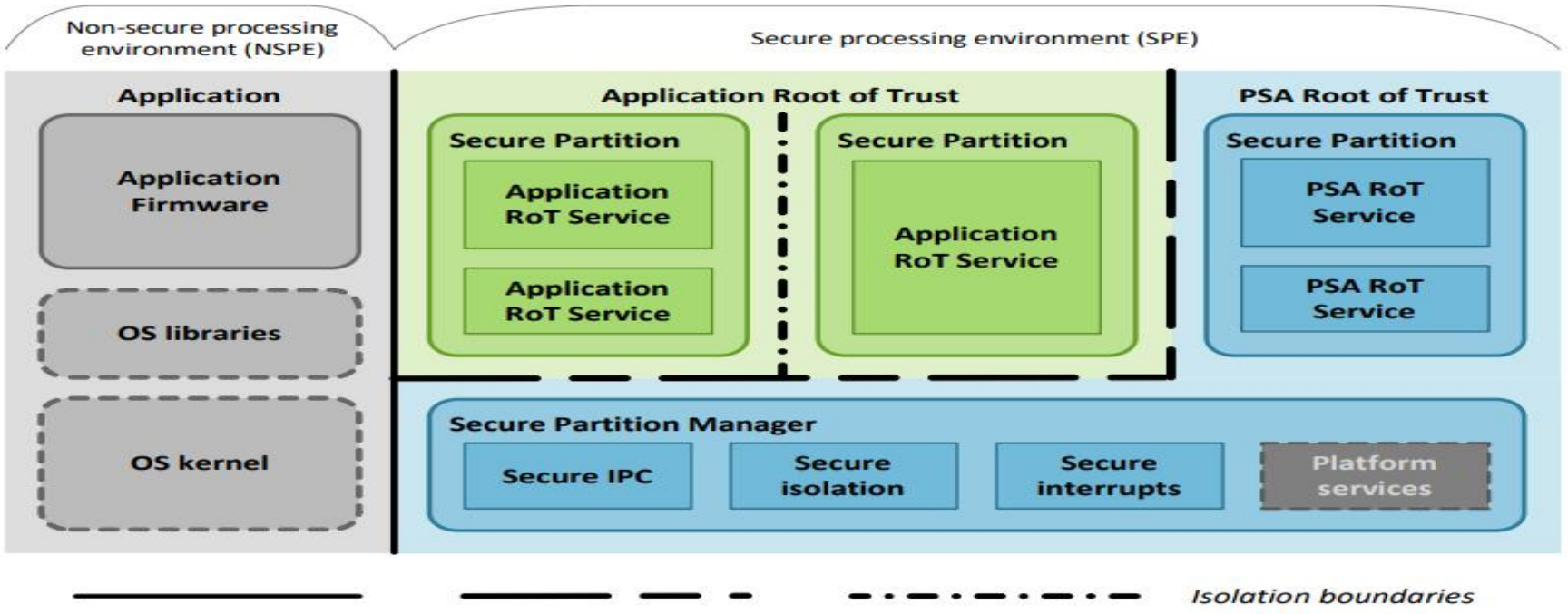
- 将系统内的执行分为两个区域-非安全处理环境（NSPE）和安全处理环境（SPE）。NSPE包含应用固件、操作系统内核和操作系统库，通常控制大多数I/O外设。SPE包含安全固件和硬件资源，与NSPE固件和非安全硬件资源隔离开来。 **(level 1 隔离)**



阶段二：构建架构规范

固件框架

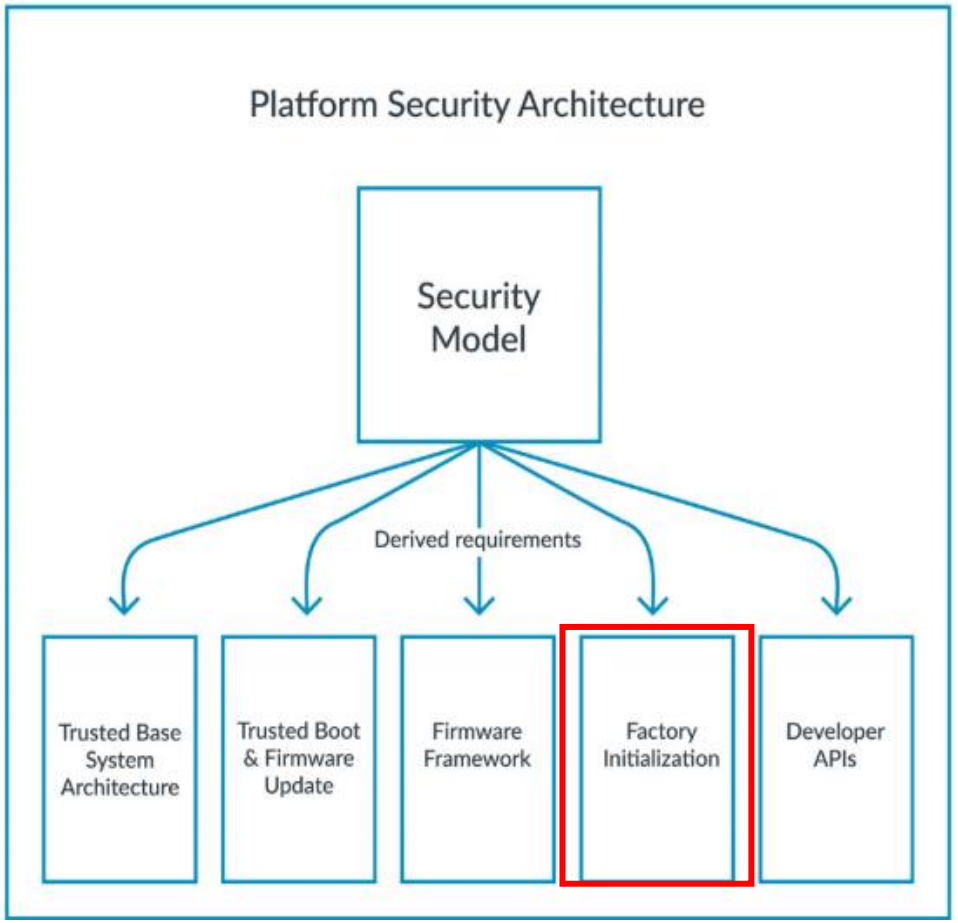
- SPE又进一步分成安全分区管理器SPM和安全分区SP（**Level 2 隔离**），由平台硬件（如Arm TrustZone）保证
- 应用可信根服务的隔离（**Level 3 隔离**）



阶段二：构建架构规范

工厂初始化

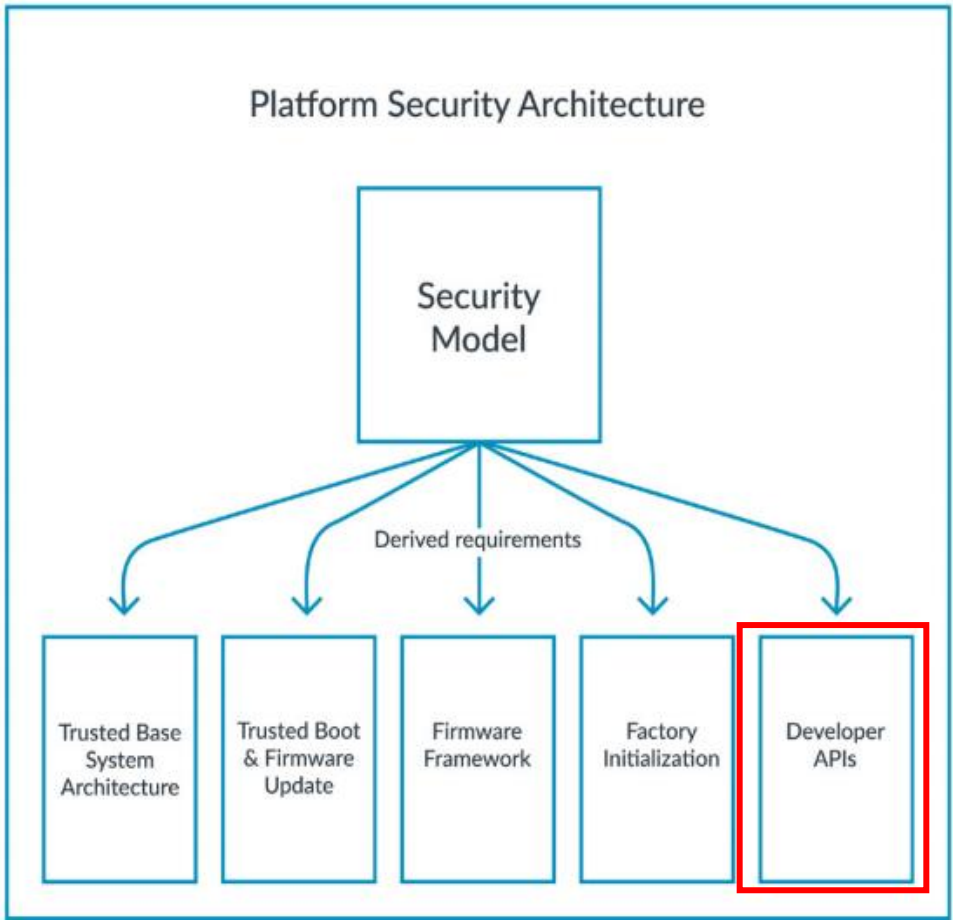
- 通过制造过程提供根密钥和初始设备固件，保证可信根（RoT）模型发挥作用
- 当构建一个安全的基础设施系统时，这个规范就显得尤为重要



阶段二：构建架构规范

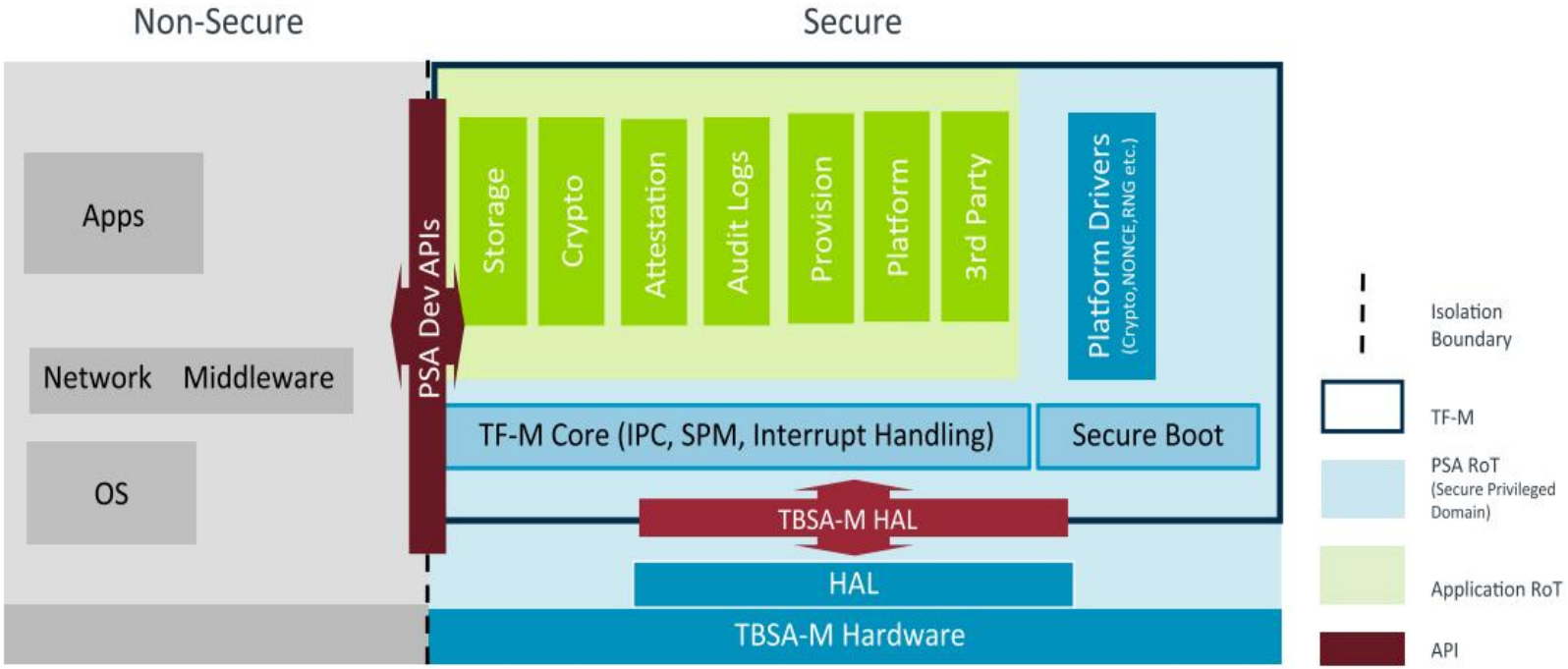
开发者API

- 加解密接口
- 安全存储接口
- 安全认证接口



阶段三：可信固件实现

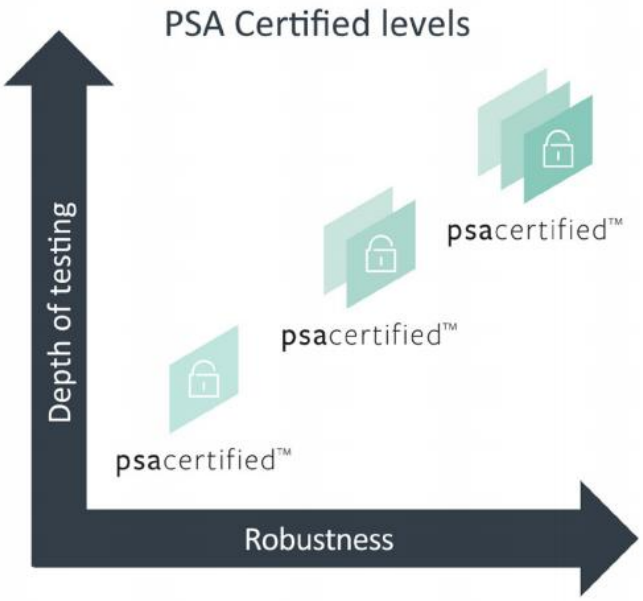
- 引导加载程序，用于安全和非安全镜像的引导，并用于镜像的安全升级
- 固件框架规范中的功能，如IPC、SPM、中断处理等
- 安全服务，如加解密、安全存储、认证、日志等
- 从非安全环境中激活安全服务的开发API
- 参考OS和开发环境，如GCC、ARMCLang编译器



阶段四：评估与认证

- 对芯片和设备在实验室条件下进行测试，评估它们的安全级别，帮助开发人员和客户相信它们可以达到所需的安全级别
  - Level 1：完成基于PSA安全模型和威胁模型的安全问卷
  - Level 2：对基于PSA可信根的实验室评估，并且开展测试软件和轻量级硬件攻击
  - Level 3：开展更广泛的攻击，例如侧信道和物理篡改等

PSA Certification level & test time	Silicon	OS	OEM
Level 3 Months	✓	3rd party evaluation schemes	
Level 2 1 month	✓		
Level 1 1 day	✓	✓	✓



内容概要

- 安全体系结构现状与不足
  - 平台安全架构（Platform Security Architecture)
  - 可信执行环境（Trust Execution Environment)
  - 存在的不足之处
- 安全体系结构框架与原理
- 总结

可信执行环境（Trust Execution Environment，TEE）

- TEE：

2010年7月，Global Platform组织首次宣布了一整套TEE系统体系标准。

标准包括一系列规范，对应用接口，应用流程，安全存储，身份认证等功能进行了规范化。当前许多商业或者开源产品一般都会参考该规范，并按照其定义的各种功能接口进行规范实现。

TEE 是一种执行环境，提供可信应用 (TA)**隔离执行**、资产**完整性和机密性保护**等安全功能。

- REE：

  富执行环境（Rich Execution Environment）包括至少一个常规操作系统和设备的所有其他组件，可以运行如Android、IOS 等通用的 OS。因此，REE也可称为**常规执行环境**（Regular Execution Environment）。

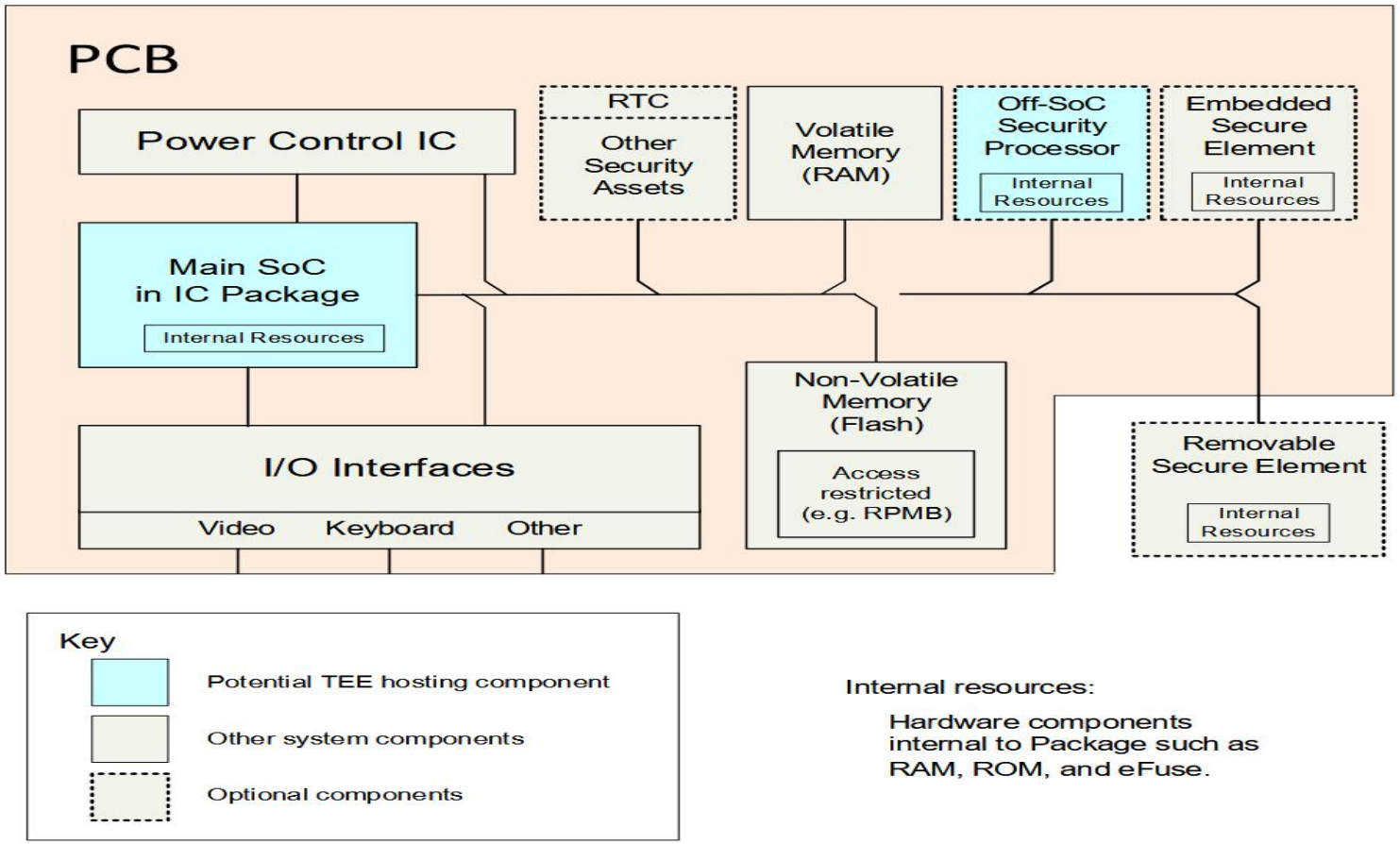
- REE & TEE：

  REE 是一个**容易受到攻击的开放环境**，从安全组件的角度来看，REE 中的一切都被认为是不可信任的。

  TEE 是一个**安全区域**，能够保证敏感数据在隔离和可信的环境内被处理，免受来自 REE 中的软件攻击。

典型的设备架构

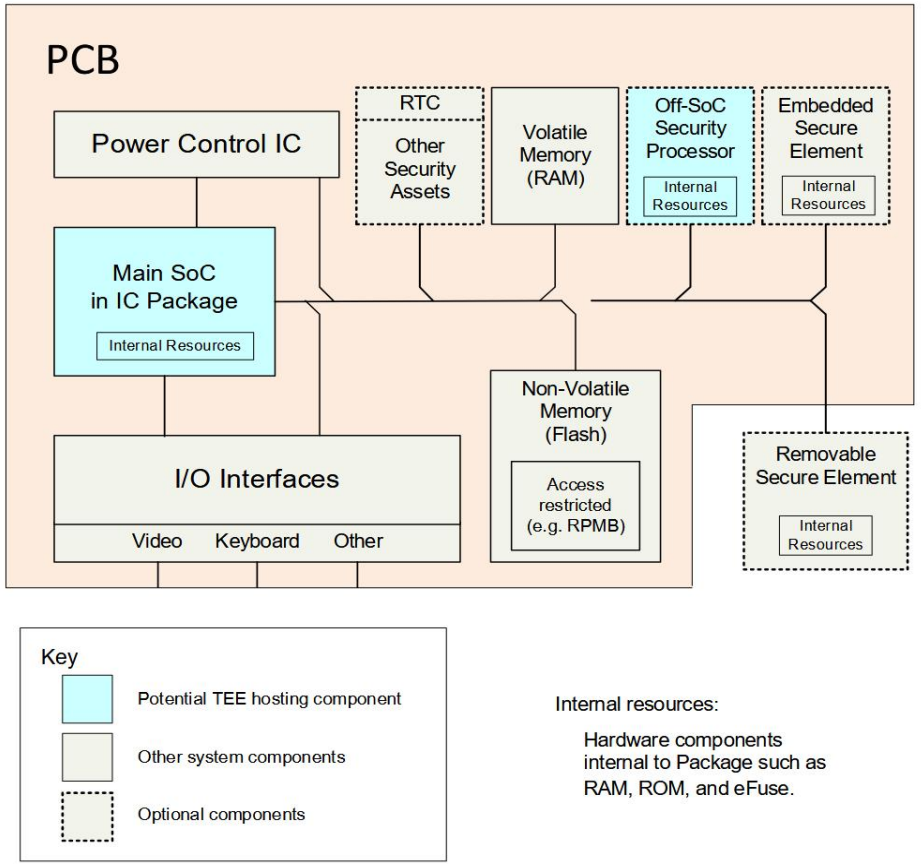
- 芯片组硬件由印制电路板(Printed Circuit Board，PCB)组成，它连接了许多组件，如SoC处理单元、RAM、Flash等。



TEE硬件架构

安全需求

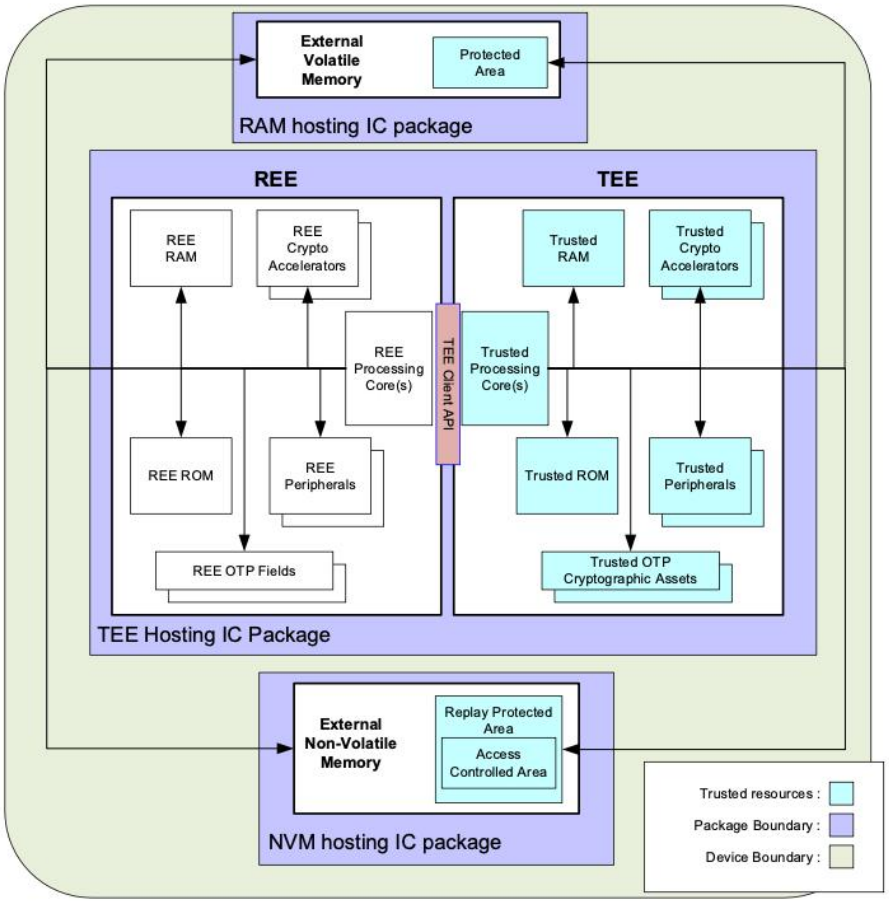
- 重要资产避免受到REE和其它环境的攻击（硬件保证）
- 避免受到某些物理攻击，如物理侧信道攻击
- 某些系统组件，如调试接口，应关闭或受到保护
- Trust OS运行时环境需由TEE中的RoT（Root of Trust）来实例化
- TEE需提供安全可信的存储和密钥的保存
- TEE外的软件不能直接调用TEE内部 API或核心框架的功能



TEE硬件架构

资源所有权

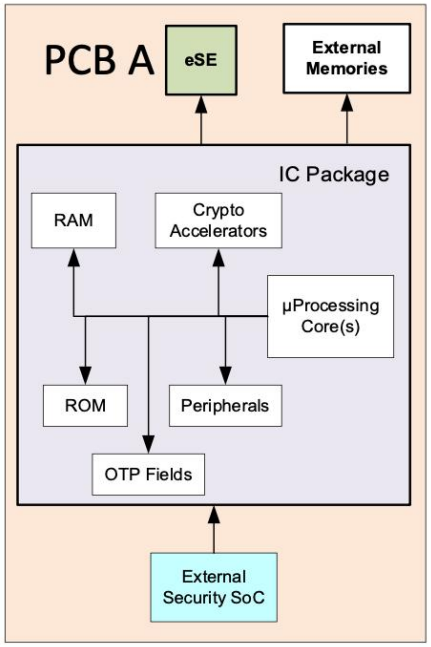
- 在任何给定时间，每个资源都由REE或TEE控制。
- 当资源由特定TEE控制时，除非由该TEE明确授权，否则REE将不能访问。
- 根据有关安全策略，TEE可以访问REE中未共享的资源。
- REE 访问受信任资源的唯一途径是通过 TEE 公开的 API 入口点或服务，并通过 TEE 客户端 API 等进行访问。

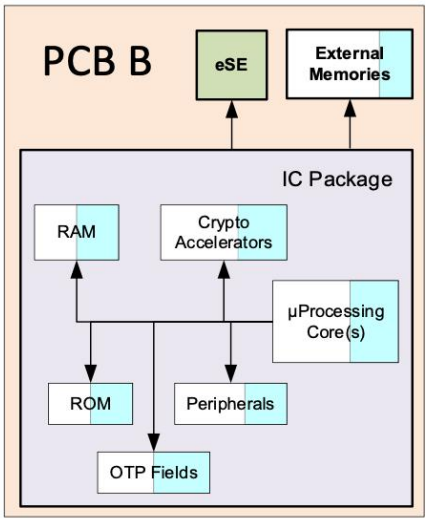


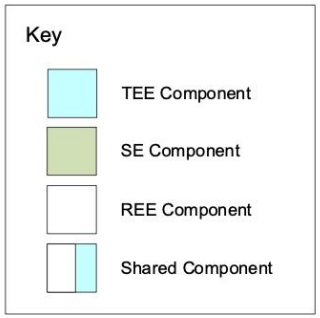
TEE硬件架构

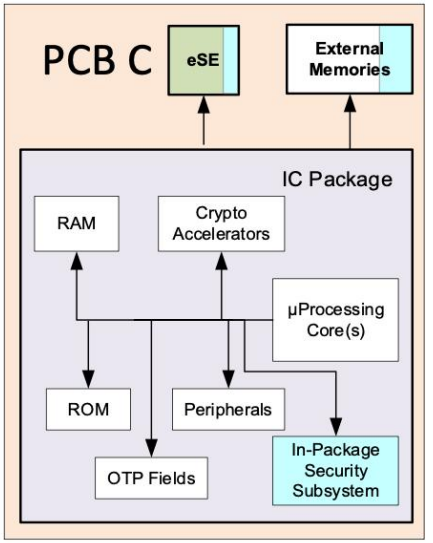
硬件实现方式

- PCB A：TEE 位于独立的安全 SoC 中，有自己的内部 RAM、ROM、内核和其他外围元件。
- PCB B：TEE和REE封装在一起，内部通过地址滤波器来隔离 RAM、ROM等资源。
- PCB C：TEE和REE封装在一起，TEE有自身内部 RAM、ROM和其他外围元件的子系统。作为可选，它还利用 eSE 和外部存储器来保存一些材料。



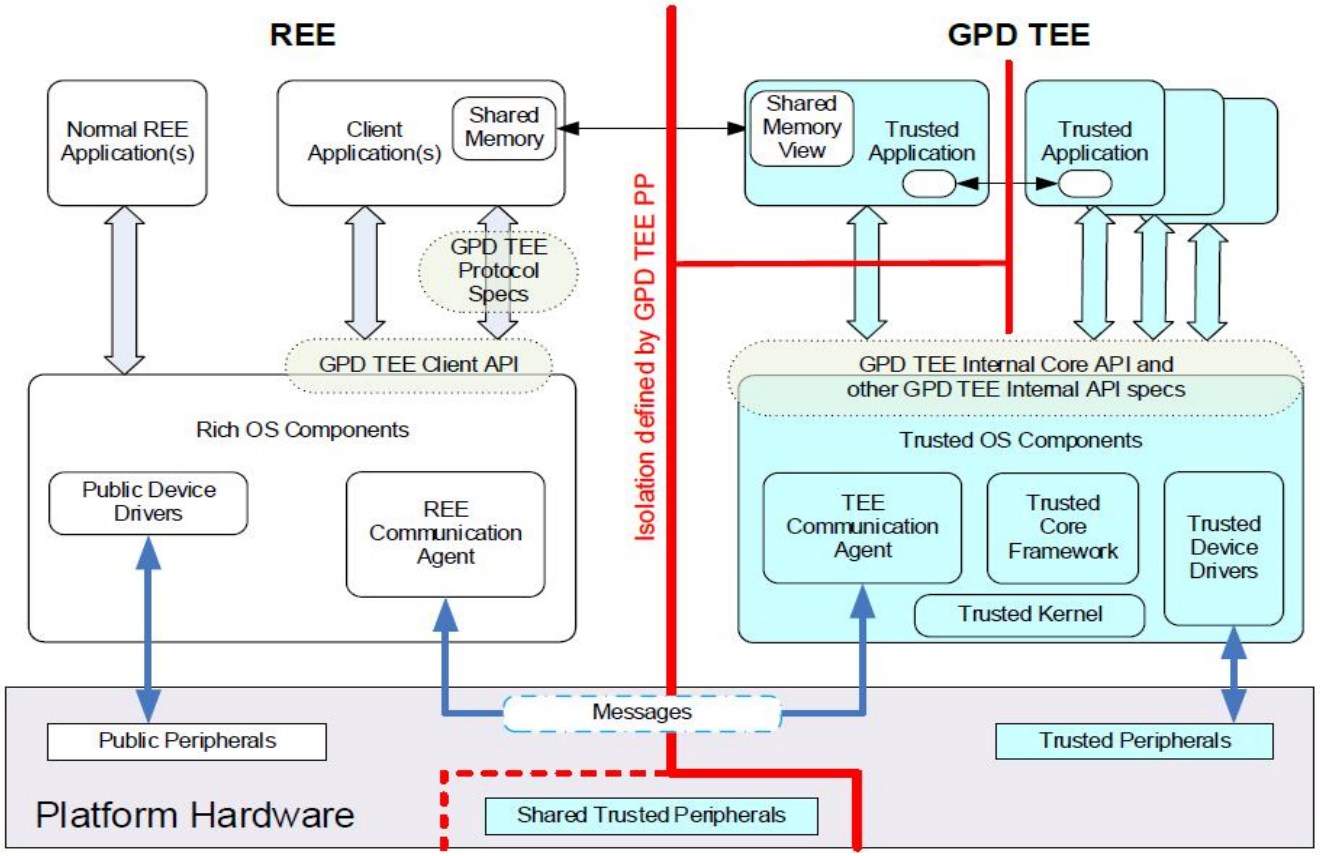






TEE 软件架构：

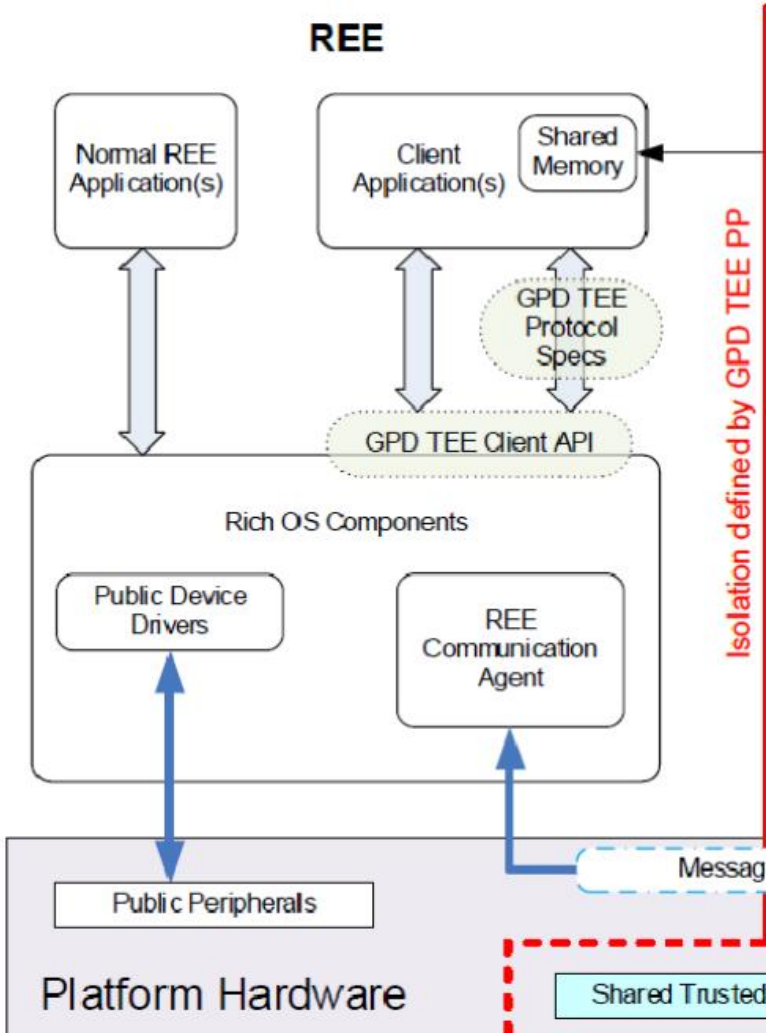
- 软件架构分为REE和TEE两部分，在REE中运行Rich OS和CA（Client APP），TEE部分运行Trusted OS和TA（Trust APP）
- 目标是使 CA 使用TA 提供安全可信的功能



TEE 软件架构

REE中的系统结构：

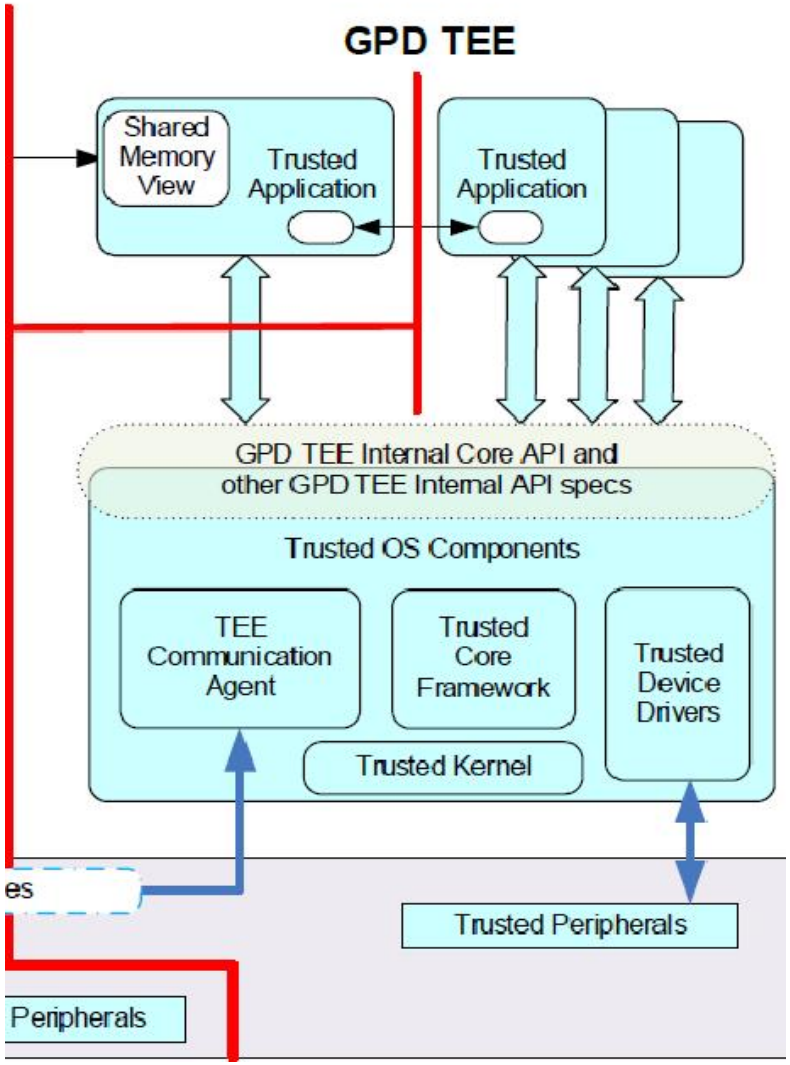
- CA对应一些上层应用，比如指纹采集、支付应用等，通过调用TEE CA实现与TEE环境的交互。
- TEE Client API是REE中的TEE驱动程序提供给外部的接口，可以使运行在REE中的CA能够与运行在TEE中的TA交换数据
- REE Communication Agent为TA和CA之间的消息传递提供了REE支持。



TEE 软件架构

TEE中的系统结构：

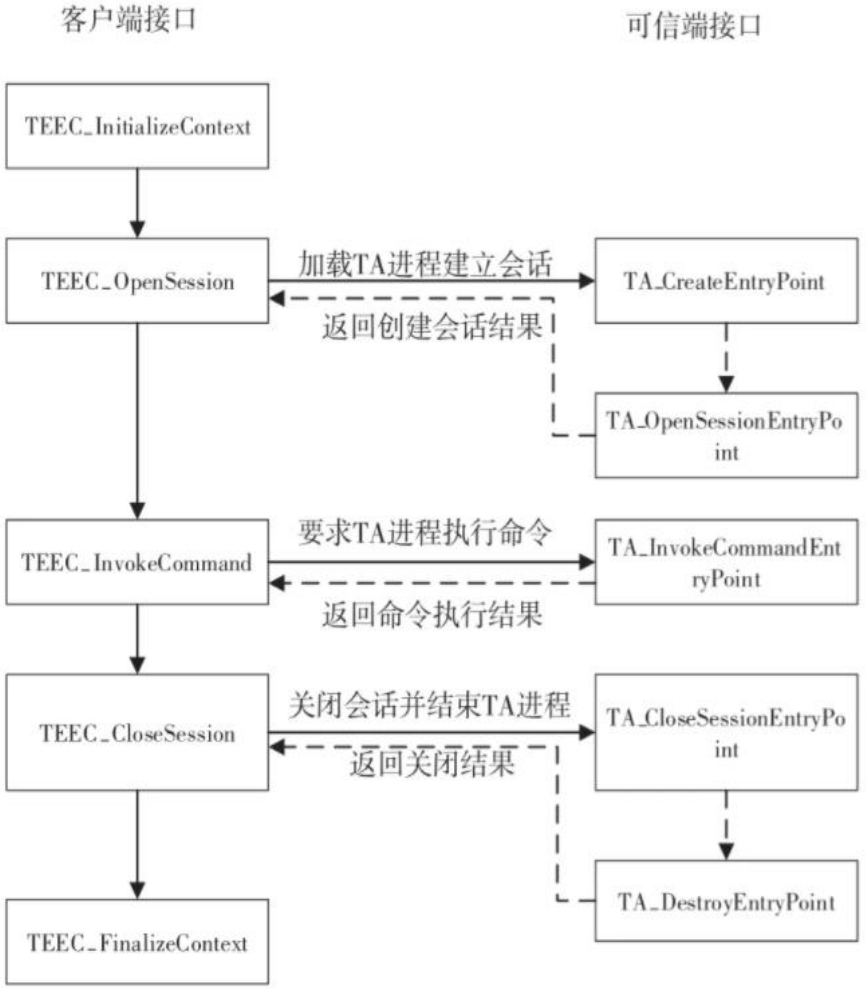
- TA是TEE中完成特定功能的应用，每个TA在REE中有一个或多个对应的CA。
- TEE Communication Agent是可信操作系统的特殊组成部分，它与REE Communication Agent一起工作，使TA与CA之间安全地传输消息。
- TEE Internal Core API是TEE操作系统提供给TA调用的内部接口，包括密码学算法，内存管理等功能。
- Trusted Device Drivers可信设备驱动程序，为专用于TEE的可信外设提供通信接口。
- Shared Memory是一块只有CA和TA可以访问的一块安全内存，CA和TA通过共享内存来快速有效传输数据。



TEE 软件架构

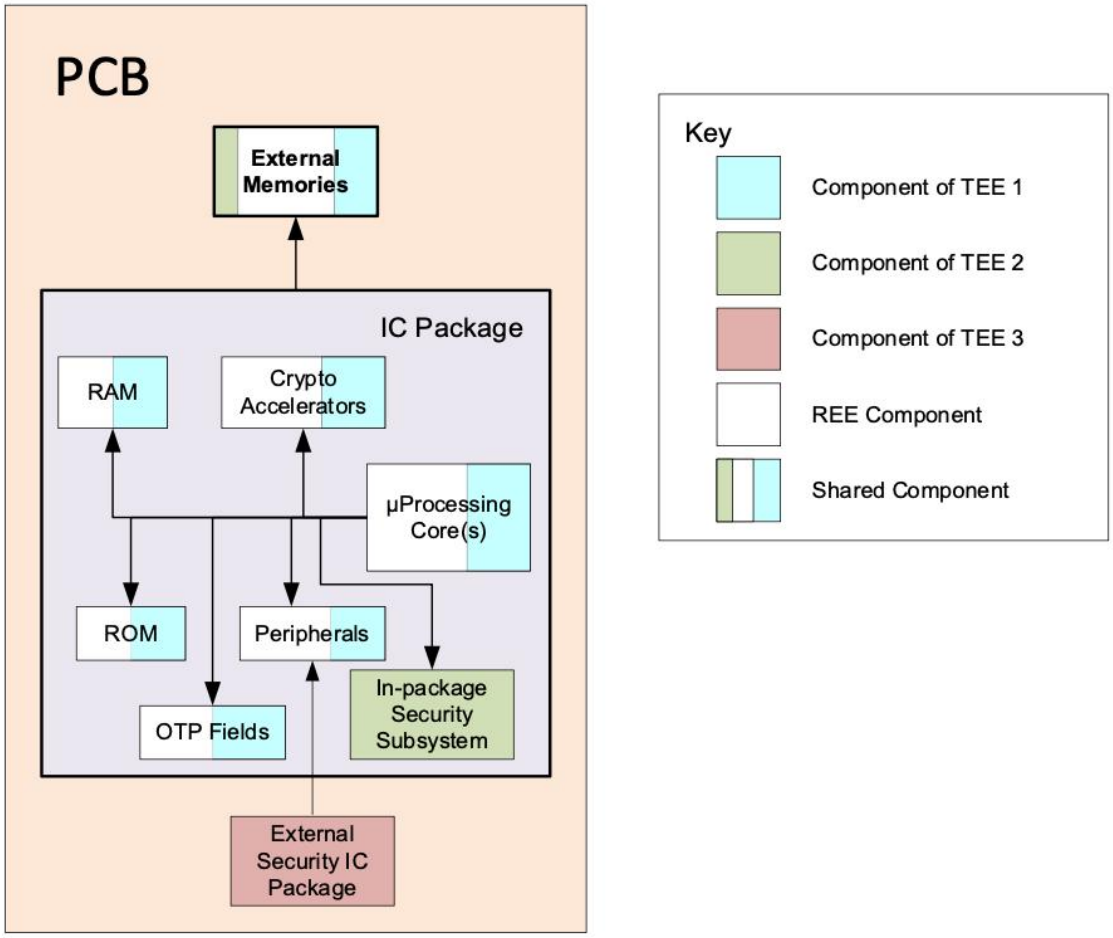
REE 与 TEE 的接口：

- REE **通信代理**为客户端应用程序和可信应用程序之间的消息传递提供 REE 支持。
- TEE **客户端 API** 是一个底层通信接口，旨在使常规操作系统中运行的客户端应用程序能够访问在可信执行环境中运行的可信应用程序并与之交换数据。
- REE 中暴露的 **TEE 协议规范层**为客户端应用程序提供了一组更高级别的 API，用于访问某些 TEE 服务。



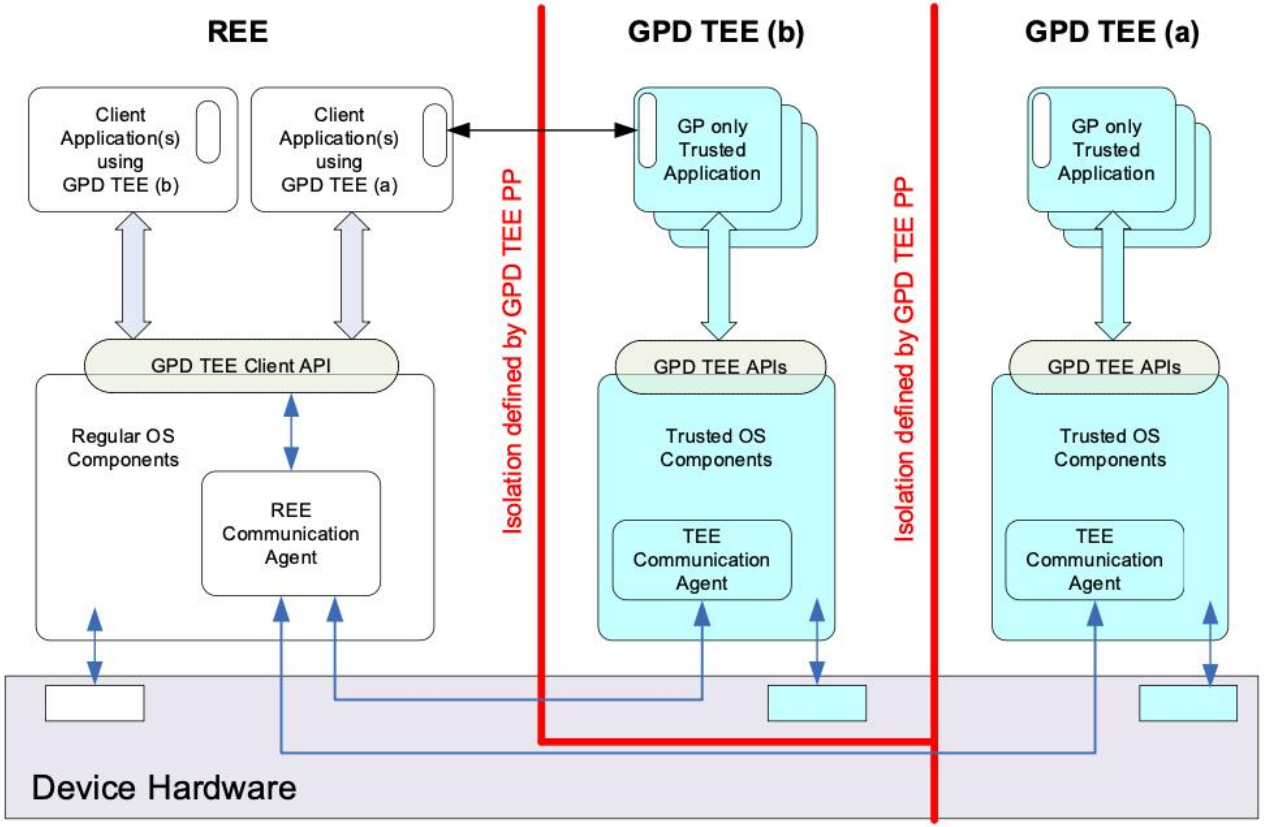
一个体系结构支持多个TEE（硬件架构）

- 图示设备有三个 TEE，每个都使用不同的示例方法创建。每个 TEE 都有一组独立的固有可信组件，并与其他 TEE 和 REE 隔离。



一个体系结构支持多个TEE（软件架构）

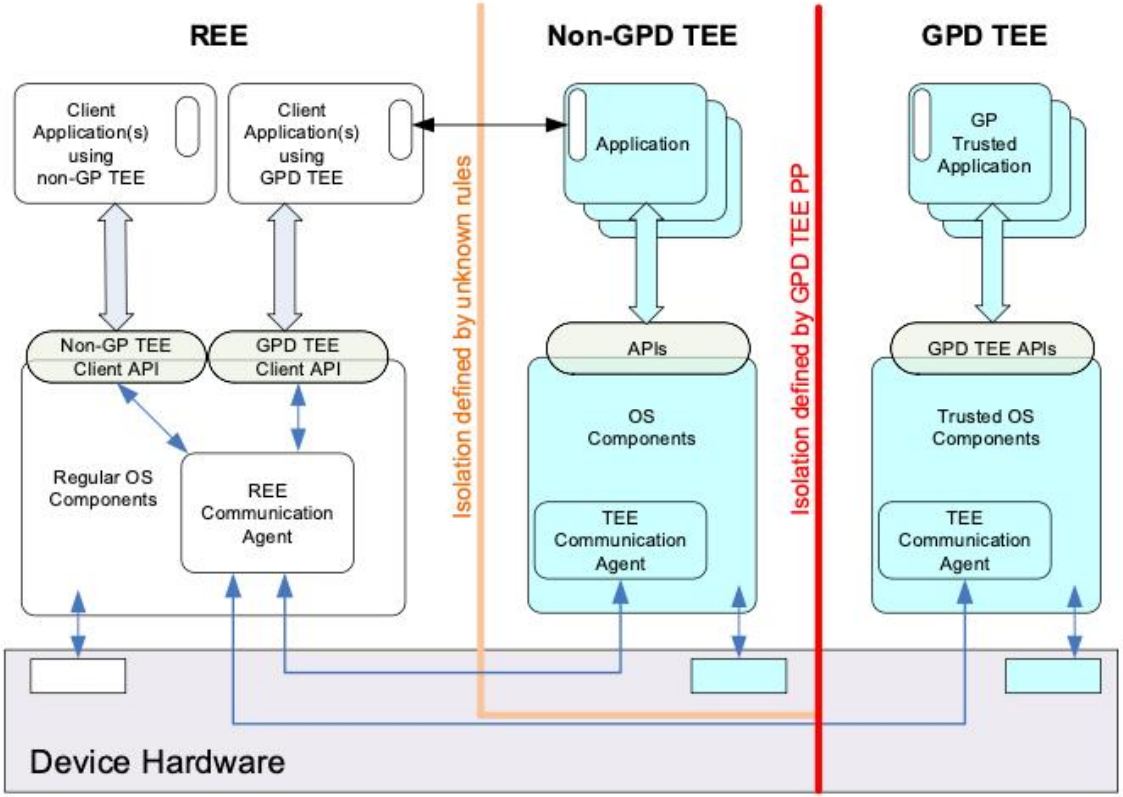
- 下图显示了两个 GPD TEE 的示例，每个 GPD TEE 都存在于自己的隔离边界内，不信任边界外的组件。因此，从 GPD TEE (a) 的角度来看，GPD TEE (b) 被认为是不可信的，因为它不是 GPD TEE (a) 的一部分。同样，GPD TEE (b) 既不信任 REE 也不信任 GPD TEE (a) 。



一个体系结构支持多个TEE（软件架构）

- 甚至可以使用声称是 TEE 但不符合 GlobalPlatform 的 TEE 的环境。该环境不会引发问题，因为符合规定的 GPD TEE 应按照 GlobalPlatform TEE 保护配置文件 ([TEE PP]) 中的规定与其隔离。

Figure 3-12: GPD TEE alongside Unknown TEE



存在的不足之处

- 只考虑了普通应用与敏感应用的运行，而**安全机制（安全应用）**该如何在系统中**存在和运行**没有涉及
- 对**抵御**由于软硬件结构设计而导致的**结构漏洞**没有提出要求
- 对**安全**与其它维度（如性能、成本）的**平衡设计**没有提出指导原则

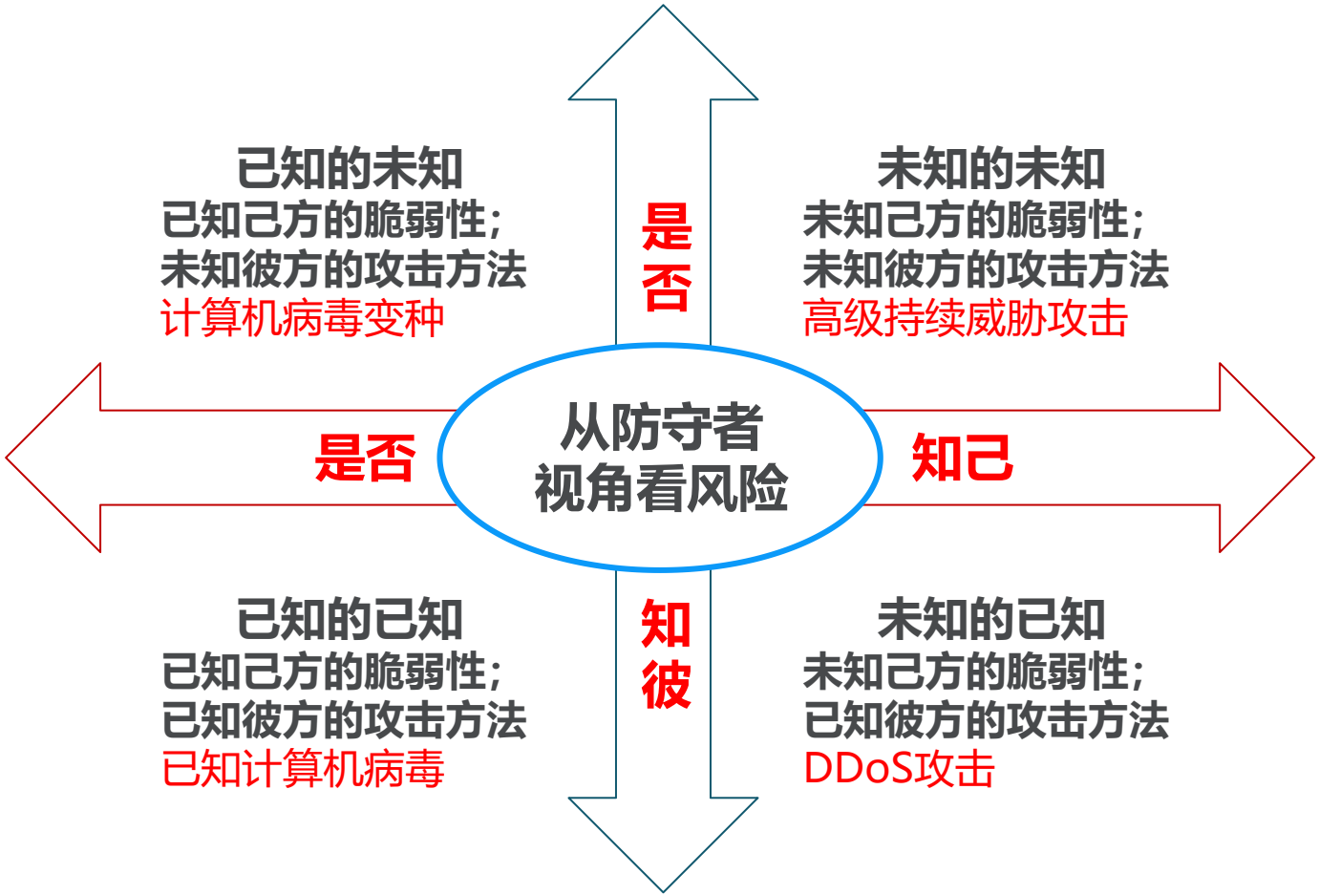
内容概要

- 安全体系结构现状与不足
- **安全体系结构框架与原理**
  - 安全体系结构概述
  - 安全体系结构组成
  - 安全体系结构内部关系
  - 安全体系结构设计原则
- 总结

安全体系结构概述

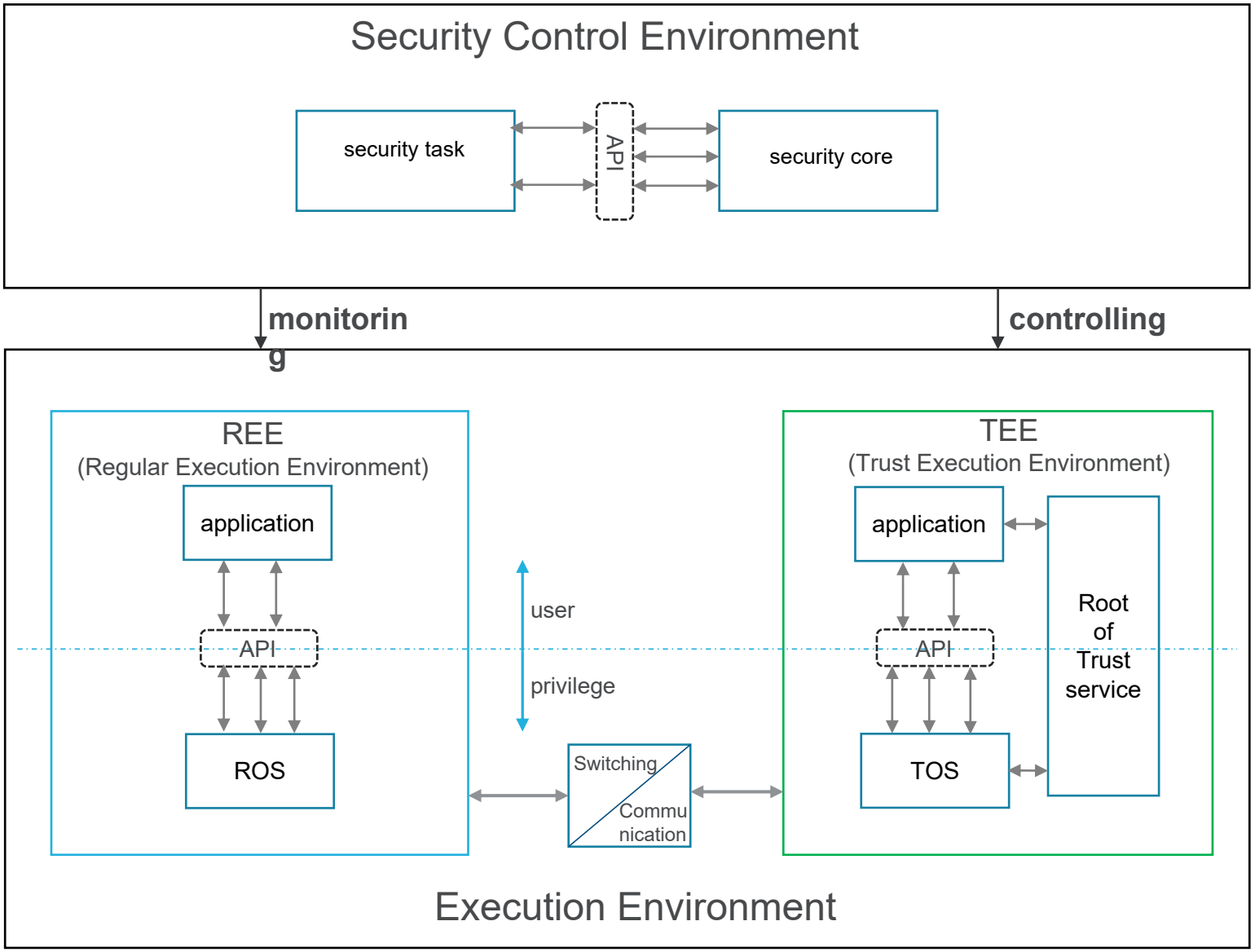
- 安全体系结构要为安全任务（安全机制）的执行赋予一个环境，以便实现**安全控制**（目标一）
- 安全体系结构要重新构建（定义）体系结构、设备和系统的**权限模型**，明确各种任务所处的权限等级（目标二）
- 安全体系结构本身要求各个结构层次（微结构、体系结构、软件结构）能**容忍脆弱性**（目标三）
- 为应对未知的威胁，安全体系结构要从结构上支持监测等**主动安全机制**（目标四）
- 安全体系结构要考虑**安全与性能的平衡**，让安全的负载不影响计算负载的性能（目标五）

未知威胁概念

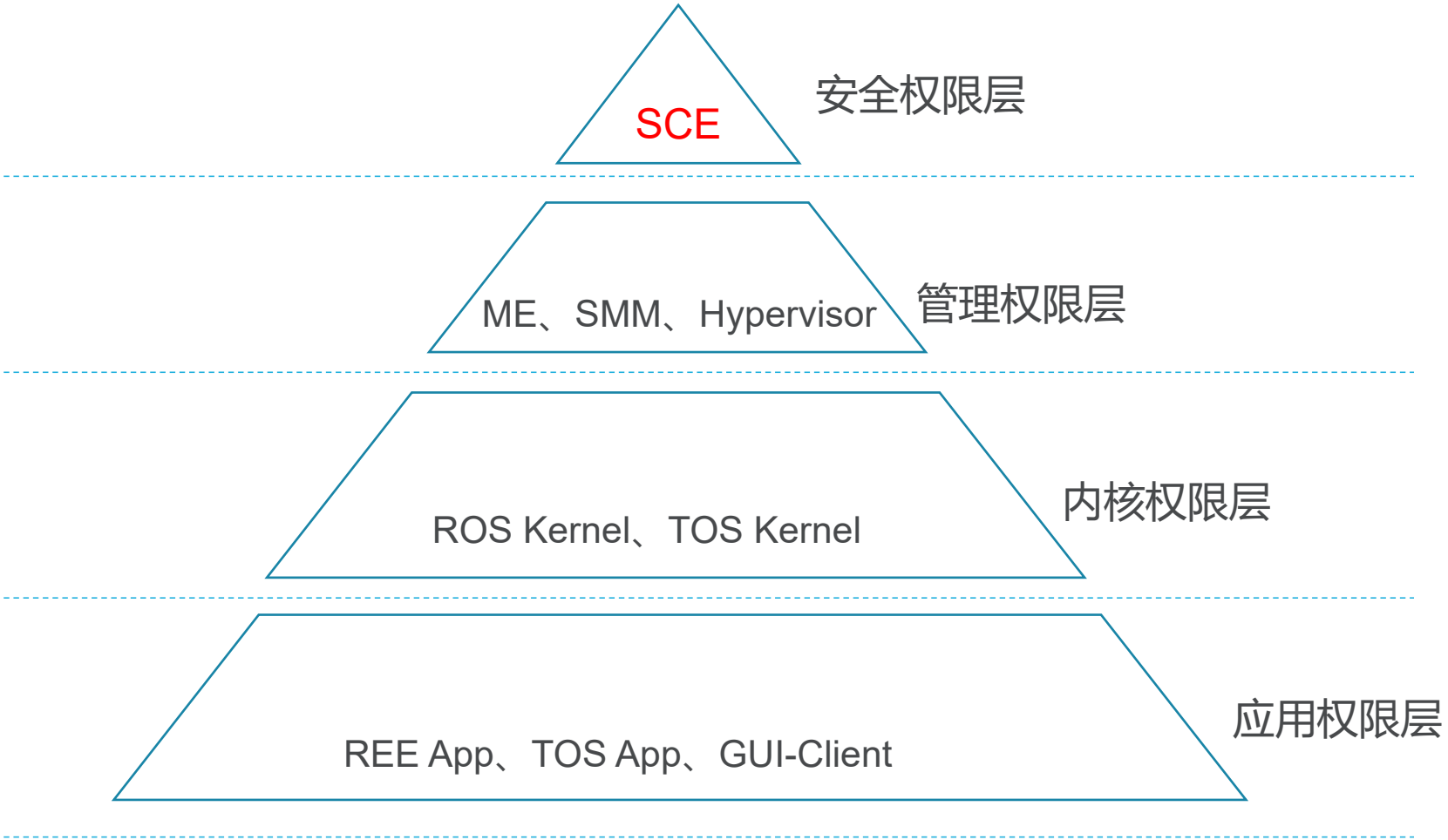


2018年，方滨兴院士在互联网安全大会的报告《从三维九空间视角重新定义网络空间安全》

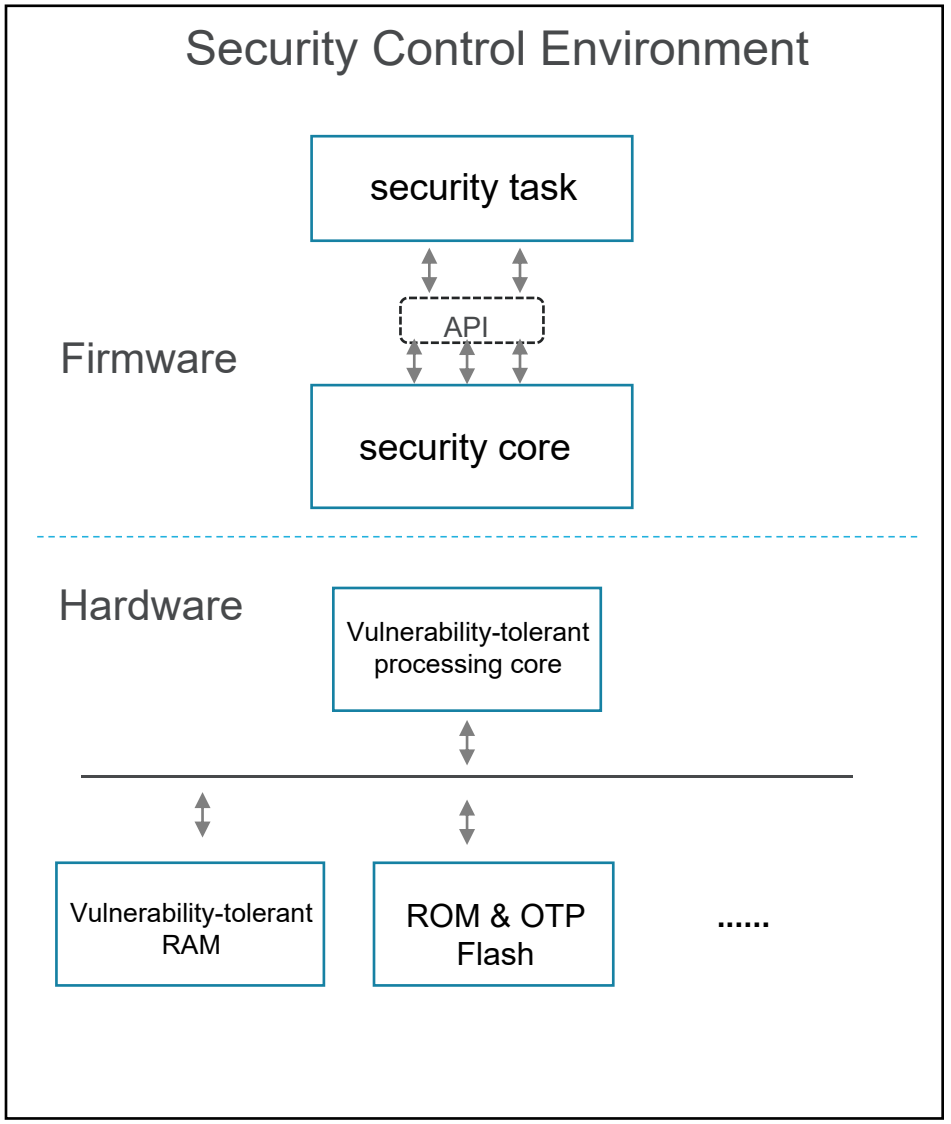
安全体系结构组成


Security Control Environment
security task
API
security core
monitoring
controlling
REE
(Regular Execution Environment)
application
API
ROS
user
privilege
Switching
Communication
TEE
(Trust Execution Environment)
application
API
TOS
Root of Trust service
Execution Environment

权限模型

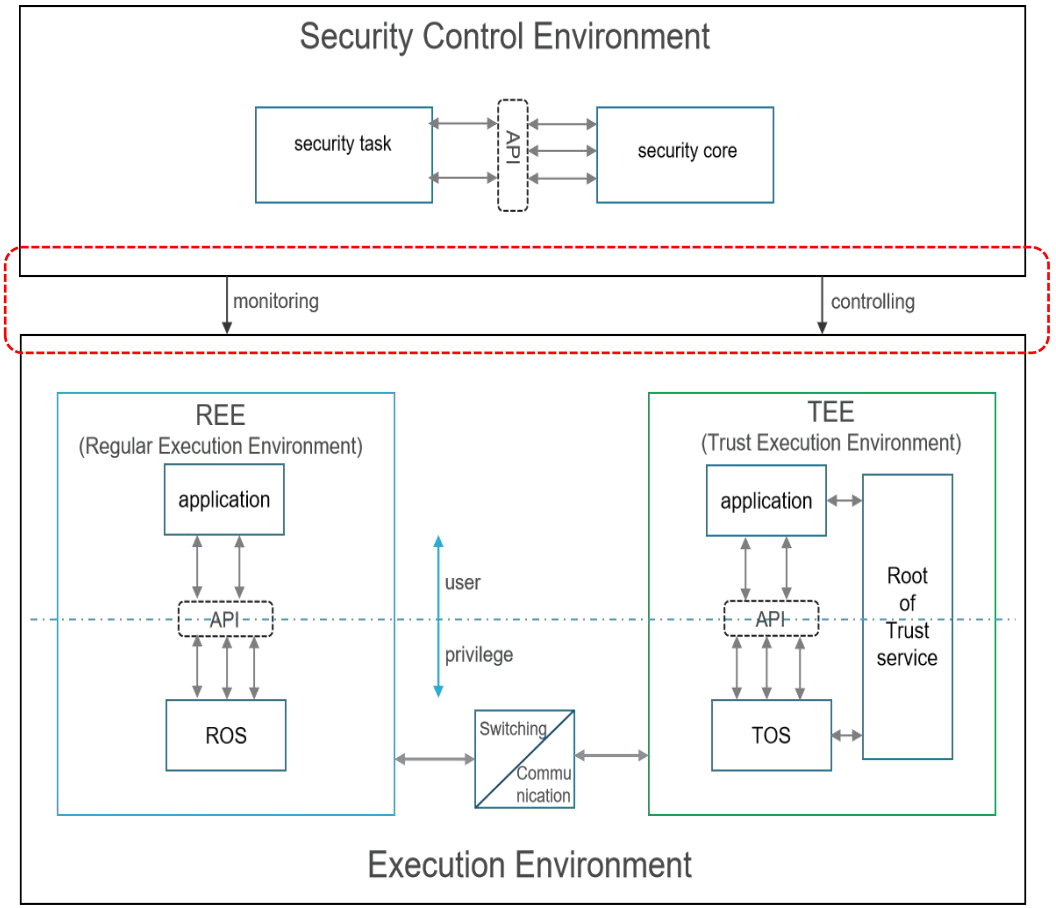

SCE
安全权限层
ME、SMM、Hypervisor
管理权限层
ROS Kernel、TOS Kernel
内核权限层
REE App、TOS App、GUI-Client
应用权限层

- 安全控制环境
  - 由硬件和固件组成，不建议出现软件
  - 硬件由脆弱性容忍的部件组成
    - 安全的处理器核
    - 安全的DRAM
    - 具有安全功能的Flash
    - .....
  - 固件包括安全核和安全任务，安全核的TCB应尽量小
  - 安全任务包括但不限于监测类、管控类、策略管理类、安全分析类等



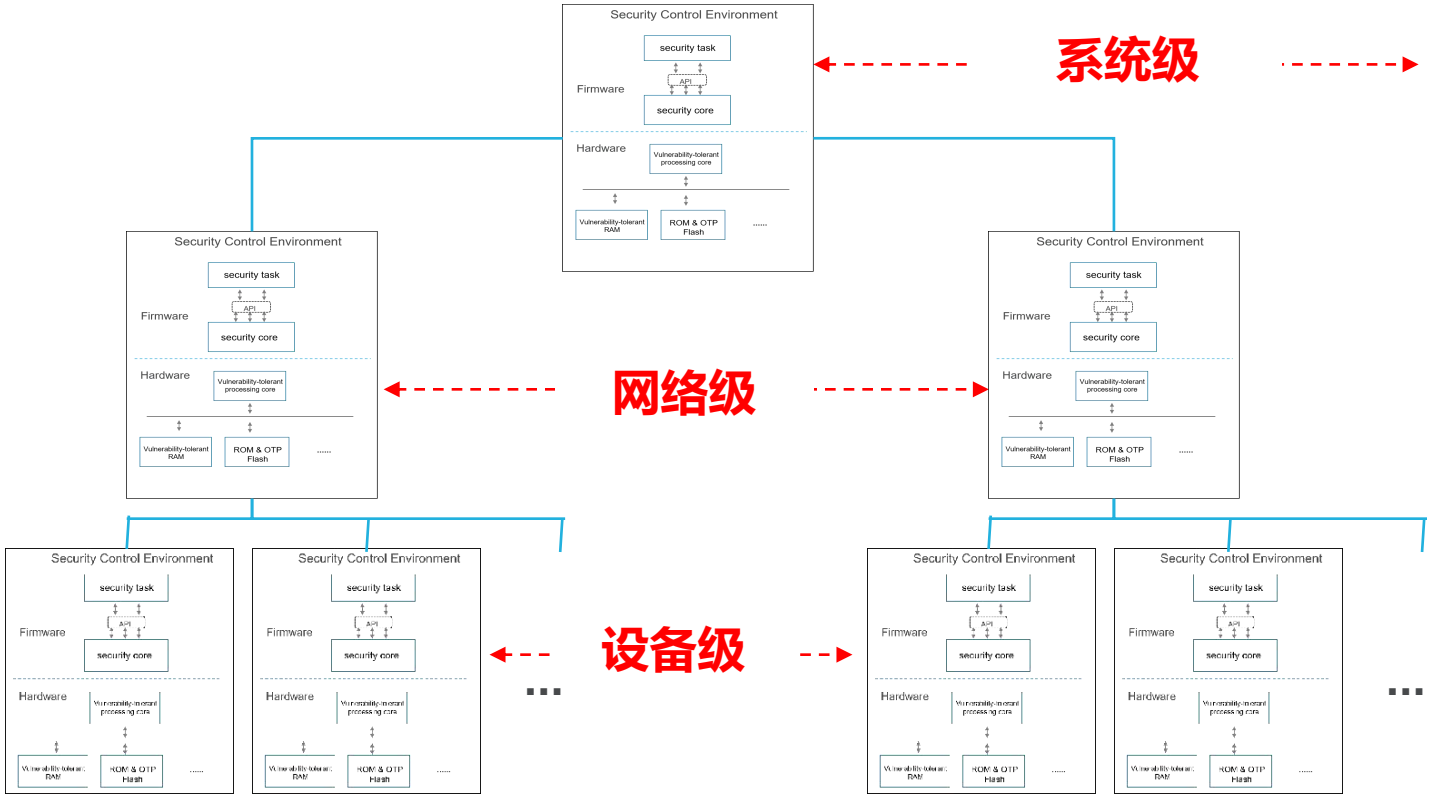
SCE与EE

- 两者通过**物理隔离**或**硬件隔离**方式确保SCE侧资源不被EE侧所访问
- SCE通过拥有的最高权限对**整个EE侧资源进行访问**，不需要通过EE的明确授权
- 系统的**初始化从SCE开始**，然后通过信任链机制传递给EE，如果EE中含有TEE，SCE将信任链传递到TEE
- SCE与EE之间的信息交互机制需要简单而明确，不**存在二义性**



SCE的级联

- SCE通过外部互连接口实现下级SCE与上一级SCE的连接
- 每一级的一个SCE都是其下一级的安全控制环境



安全体系结构设计原则

- 独立性原则
- 隔离性原则
- 最小特权原则
- 最小攻击面原则

- 独立性原则
  - SCE需要有独立的计算和存储资源，从而实现“目标五”、“目标一”和“目标二”
    - 不抢占EE环境的资源，对EE任务影响小
    - SCE独立运行，不受EE控制，是实现对EE控制的必要条件
  - SCE独立形成互连网络，在确保SCE最高权限的情况下，防止其它网络对SCE网络的攻击
  - SCE独立且率先启动，可以保证SCE作为RoT，并以此为基础，构成完整的信任链

- 隔离性原则
  - SCE与EE采用强隔离的方案，从而实现“目标五”和“目标二”
    - 逻辑隔离方案，如加密隔离，仍然会出现挣资源的情况
    - 通过强隔离，消除权限提升的途径，为SCE拥有最高权限提供了必要条件
- 最小特权原则
  - 只赋予SCE完成监测和调控所需的必要权限，从而实现“目标一”和“目标四”
    - 需对监测的目标和粒度进行分析和确定，以便依此设计相应的权限
    - 需对调控的对象和方式进行分析和确定，以最简单的规则，最小的干预，实现控制

- 最小攻击面原则
  - SCE与EE只能通过确定的方式、确定的语义和松耦合方式进行通讯
    - 简单、异步、确定的通信机制
  - SCE中的安全核（Security Core）及安全任务要精简，防止通过SCE网络或其它途径的软件攻击
    - 形式化证明的安全核
  - SCE中的组件，应采用结构安全的部件，从而实现“目标三”
    - 微结构安全的处理器核

内容概要

- 安全体系结构现状与不足
- 安全体系结构框架与原理
- **总结**

内容概要

- 介绍了ARM提出的PSA安全体系结构
- 介绍了可信执行环境（TEE）规范
- 讨论了安全体系结构的设计目标、结构组成、相互关系和设计原则

作业

1. Intel处理器目前的权限模型在第一讲中提到，在不改变Intel处理器权限模型的条件下，能否实现本节课所述的安全体系结构?如果不能，请说出理由；如果能，描述权限如何设计?

2. 阅读参考文献：

《Arm_Platform_Security_Architecture_Overview_WhitePaper》

《GlobalPlatform Technology TEE System Architecture v1.3》


Q&A