SMART: Secure and Minimal Architecture for (Establishing a Dynamic) Root of Trust

SMART vs OP-TEE scope

- **SMART** is designed for **low-end microcontrollers (AVR, MSP430, Cortex-M)** that *lack* security features like an MMU/MPU or TrustZone. It proposes *minimal hardware modifications* (ROM-resident code, secret key storage, reset/erase logic) to create a **dynamic root of trust**.

- **OP-TEE**, on the other hand, already assumes you are on an ARM processor with **TrustZone**. It provides a **Trusted Execution Environment (TEE)** with isolated Secure World and can support secure boot and attestation.

Dynamic Root of Trust
SMART's key contribution is enabling **dynamic attestation at runtime** even if the device is compromised.
OP-TEE generally relies on **secure boot (static root of trust)** + TrustZone isolation. Dynamic attestation (like SMART) is not a built-in primitive in OP-TEE, though you can build remote attestation frameworks on top of it.

Hardware modifications
SMART *requires changes to the MCU hardware* (special ROM access control, key isolation, memory wipe on reset).
OP-TEE **cannot change hardware**, it only runs on top of what ARM TrustZone gives.

So is SMART possible in OP-TEE?

- If your target is **ARM TrustZone-based hardware (Cortex-A or Cortex-M with TZ)**, you don't need SMART. OP-TEE already provides trusted ROM, secure key storage, and isolation. You can build attestation protocols similar to SMART on top of OP-TEE (using secure storage + TAs).

- If your target is **low-end MCU without TrustZone**, OP-TEE is **not even portable** there, but SMART can be

On **ARM TrustZone systems**: SMART is unnecessary, OP-TEE already provides equivalent or stronger primitives.

On **low-end MCUs without TZ**: OP-TEE cannot run, but SMART works.

.