eBPF and Confidential Computing

Dave Thaler <dthaler@microsoft.com>

Two foundations under Linux Foundation

- Confidential Computing Consortium https://confidentialcomputing.io/
- eBPF Foundation https://ebpf.foundation/
- Members in common:
  - Google, Huawei, Intel, Meta, Microsoft, Red Hat
  - 6/9 eBPF Foundation premier members
  - 6/8 Confidential Computing Consortium platinum members

WHAT IS EBPF?

*eBPF is a **cross-platform** technology*

that can run sandboxed programs

*to extend a **privileged system component***

eBPF runs in many contexts

Multiple operating systems

Kernel mode or user mode

Main processor, co-processor, SmartNIC, etc.

Host or Guest VM

Inside or outside containers

Native (JIT-compiled) code or interpreted

All using common toolchains and APIs

Typical scenarios

Design time:

- **Design an extension to be deployed it into an existing runtime environment (or even included with environment distribution)**

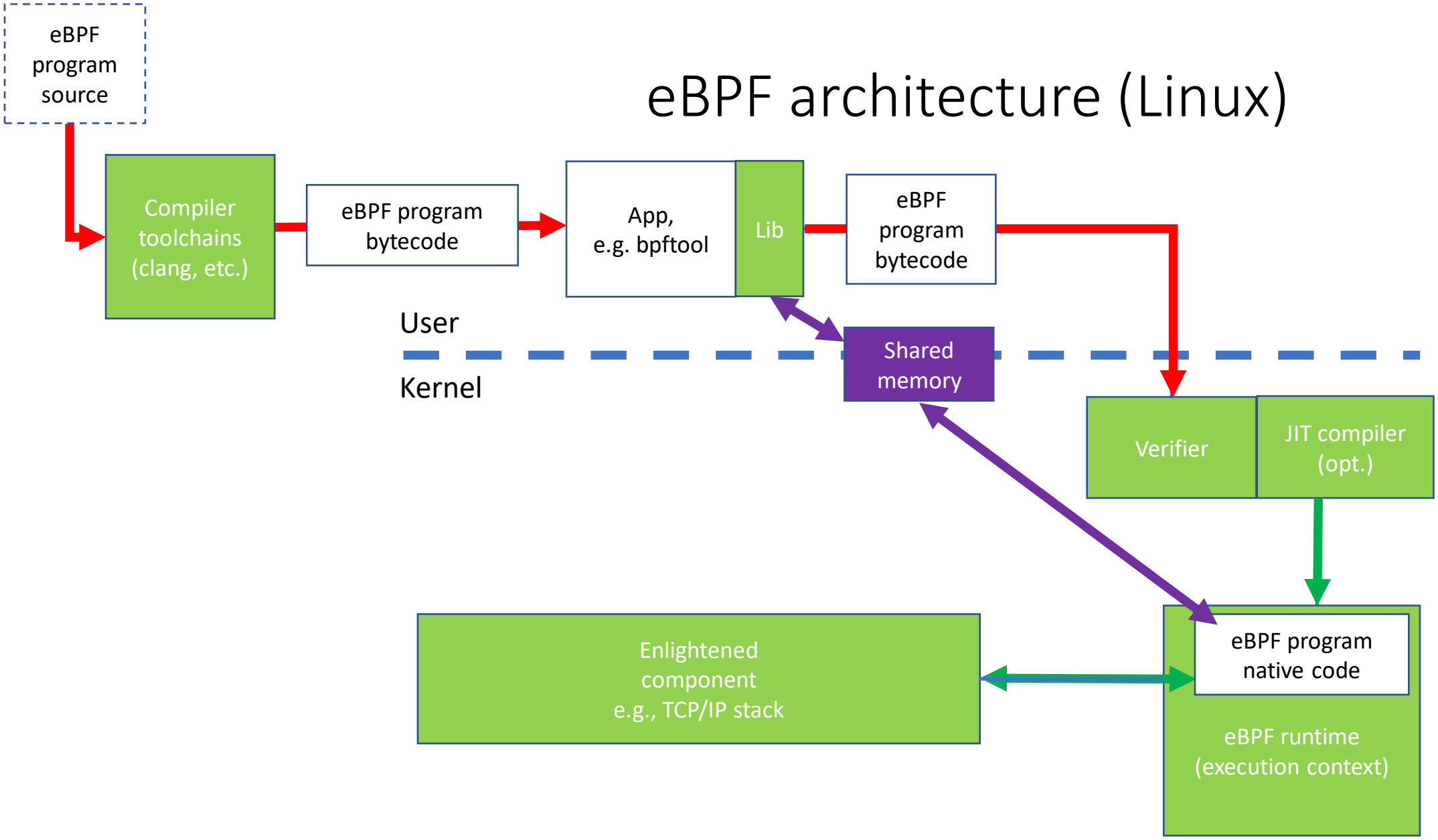
- Examples:
  - NAT
  - Telemetry/logging tool
  - file system redirector
  - Custom security policy

Run time:

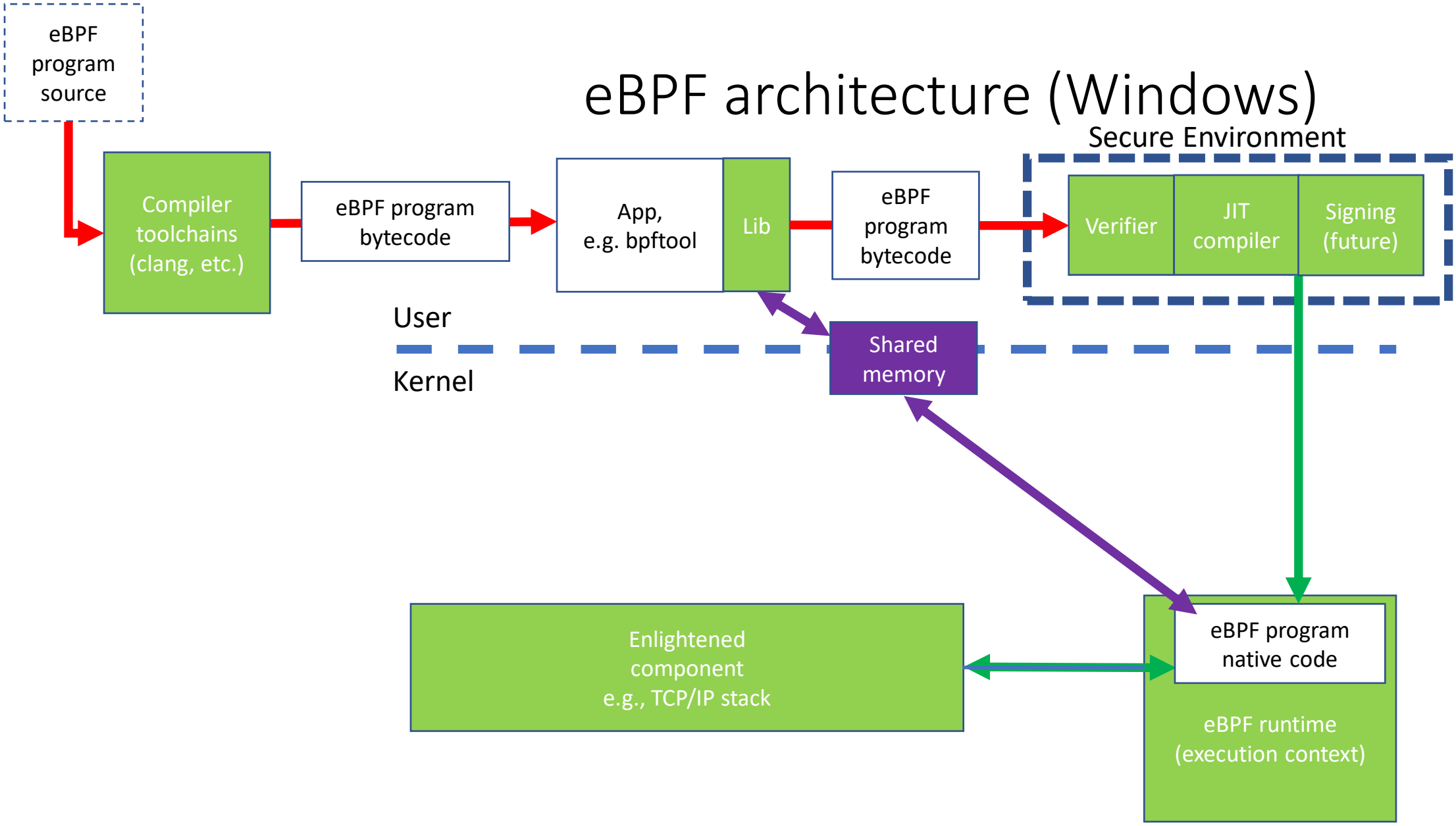
- **Create an extension on the fly**

- Examples:
  - admin-typed filter for observability
  - mitigate a DOS attack

eBPF architecture (Linux)



eBPF architecture (Windows)



What is Confidential Computing?

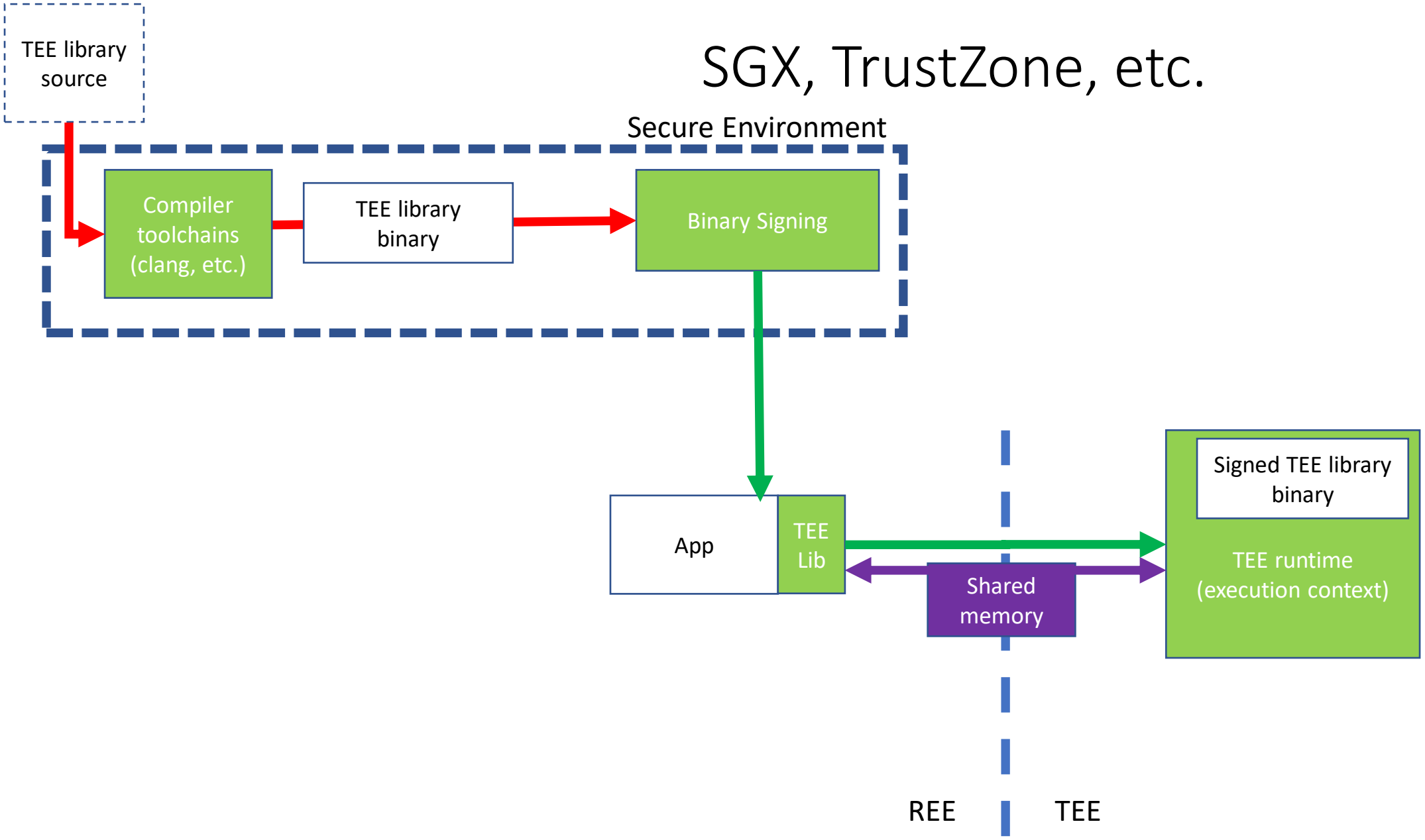
*Confidential Computing is the protection of data in use by performing computation in a **hardware-based**, **attested Trusted Execution Environment**.*

What is Confidential Computing?

*Confidential Computing is the protection of data in use by performing computation in a **hardware-based**, **attested Trusted Execution Environment**.*

*A Trusted Execution Environment (TEE) is an environment that enforces that provides a level of assurance of **data integrity**, **data confidentiality**, and **code integrity**.*

SGX, TrustZone, etc.

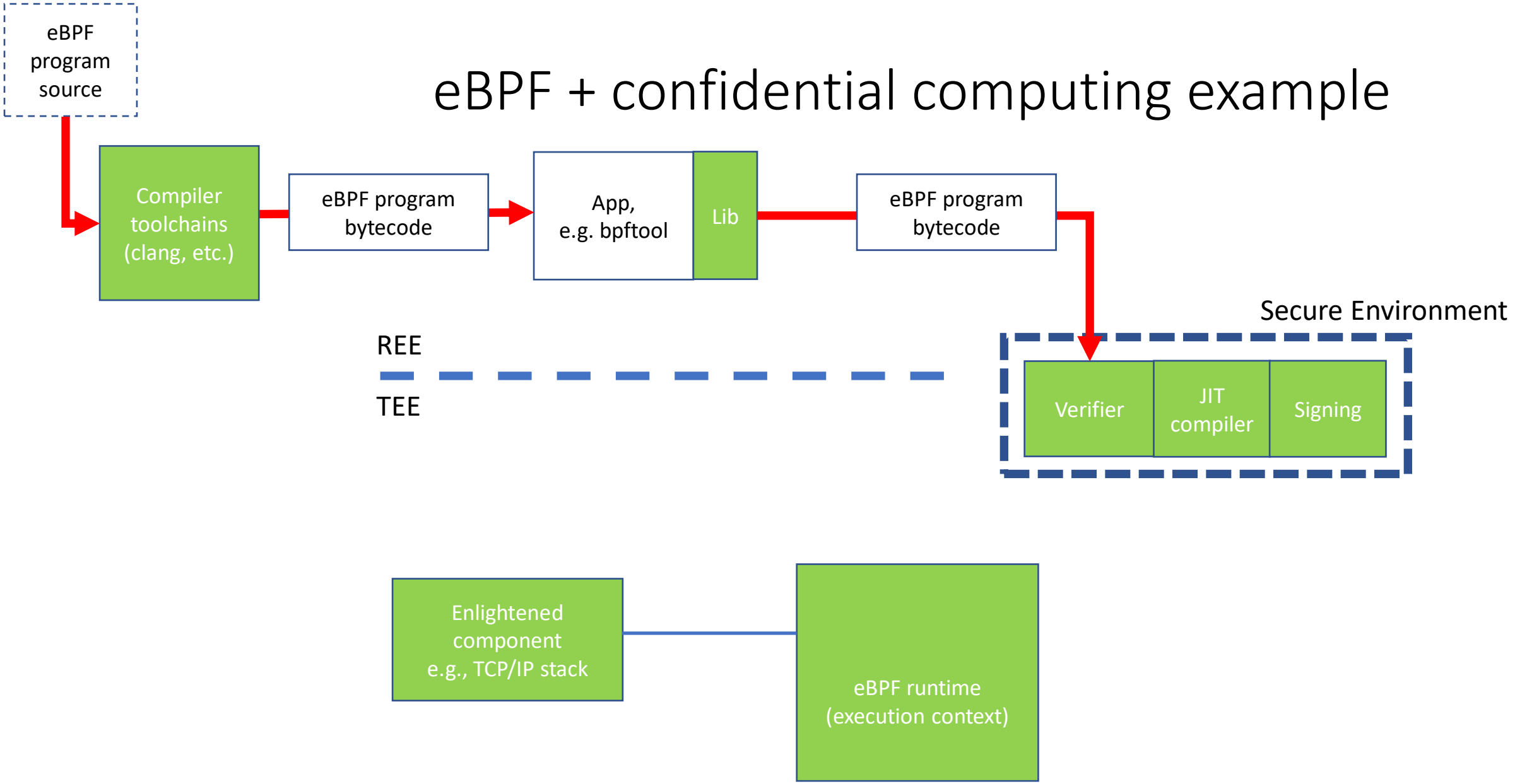


Putting them together

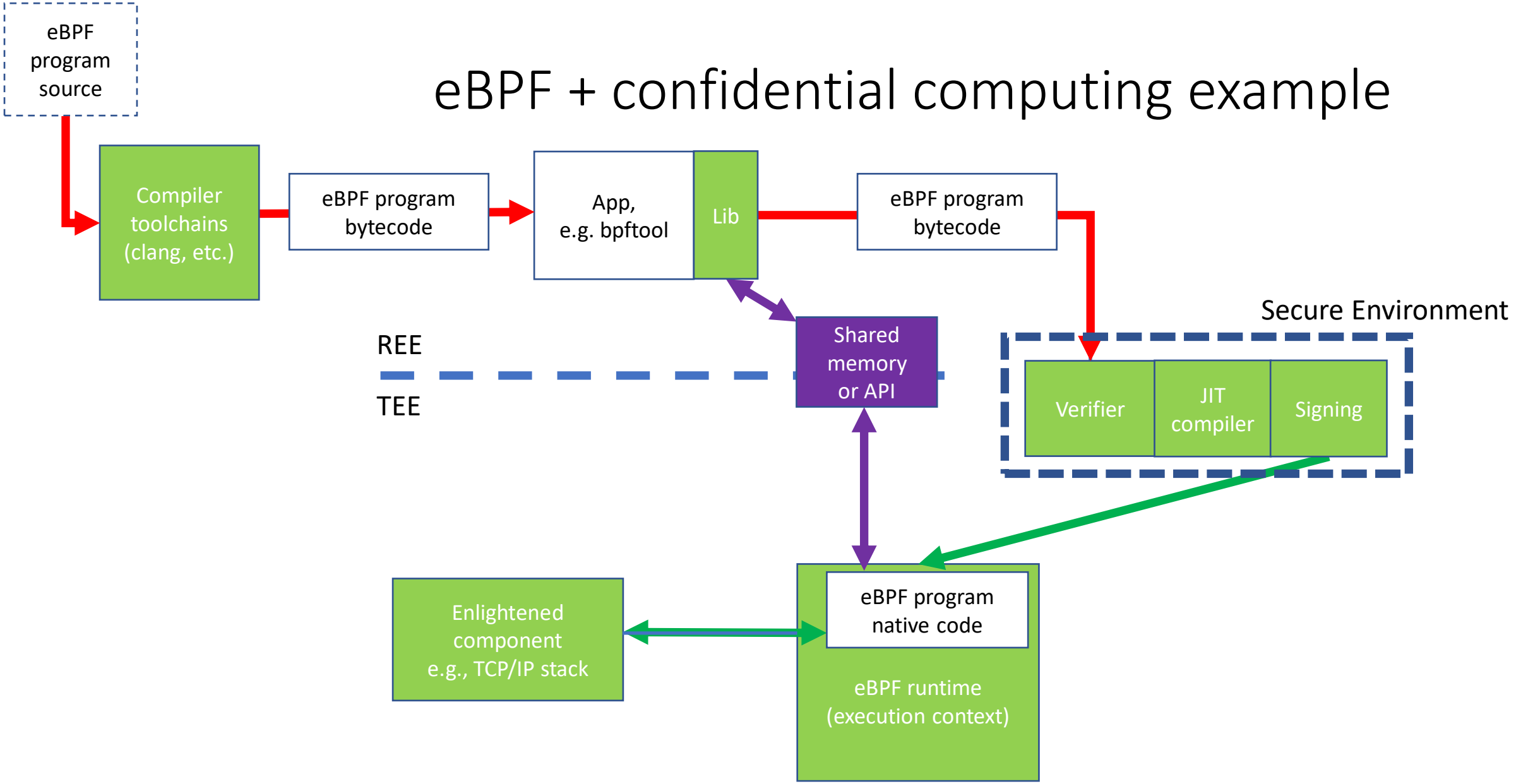
- *"eBPF is a cross-platform technology that can run sandboxed programs to extend a **privileged system component**"*
  - ➢ Code in a TEE is a privileged system component
  - ➢ Like a SmartNIC in that it's not part of the normal CPU REE

- Both scenario types still apply:
  - **Design-time scenario:** Design an extension to be deployed it into an existing confidential VM/container/process/library
  - **Run-time scenario:** Create an extension on the fly based on admin input, to run in an existing confidential VM/container/process/library

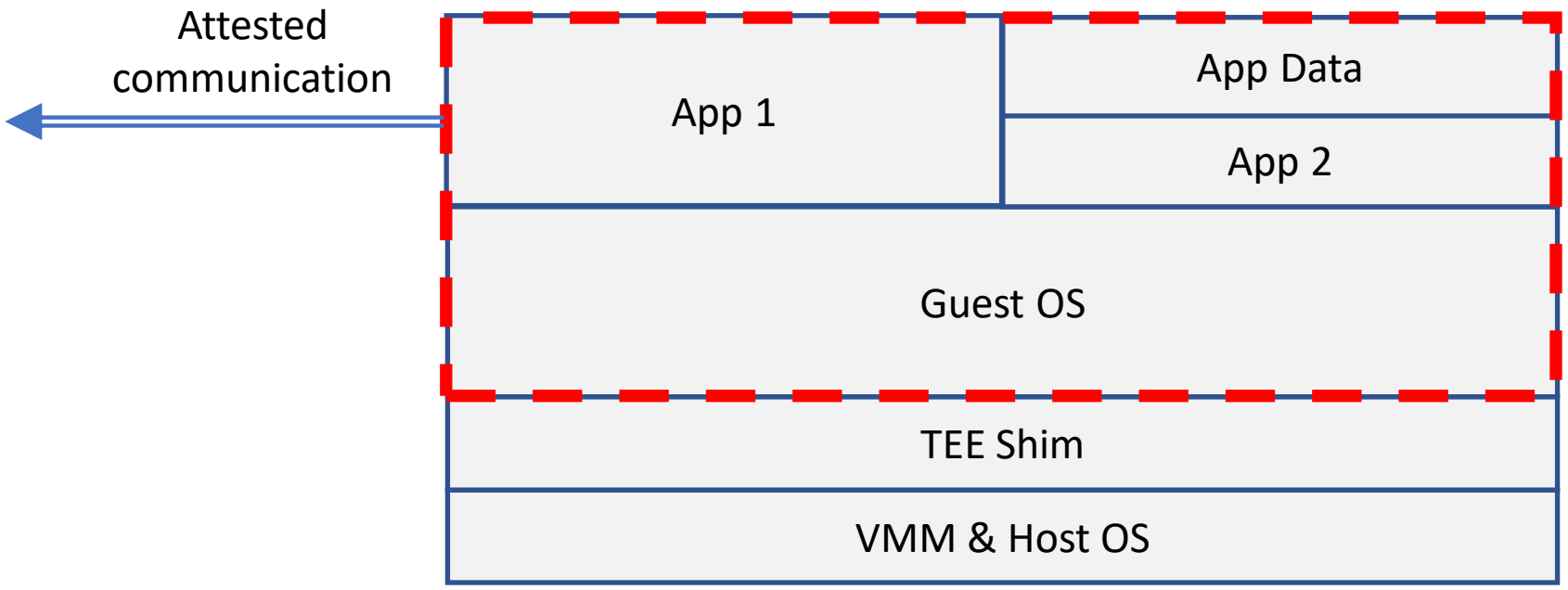
eBPF + confidential computing example



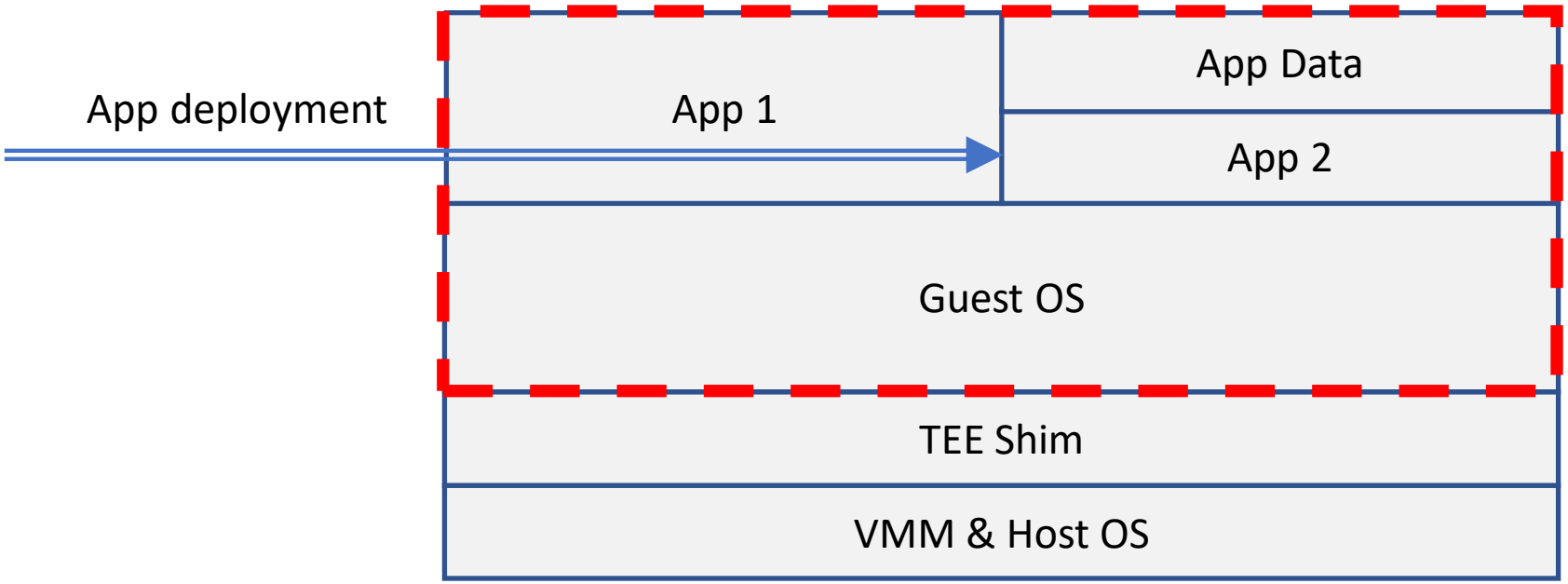
eBPF + confidential computing example



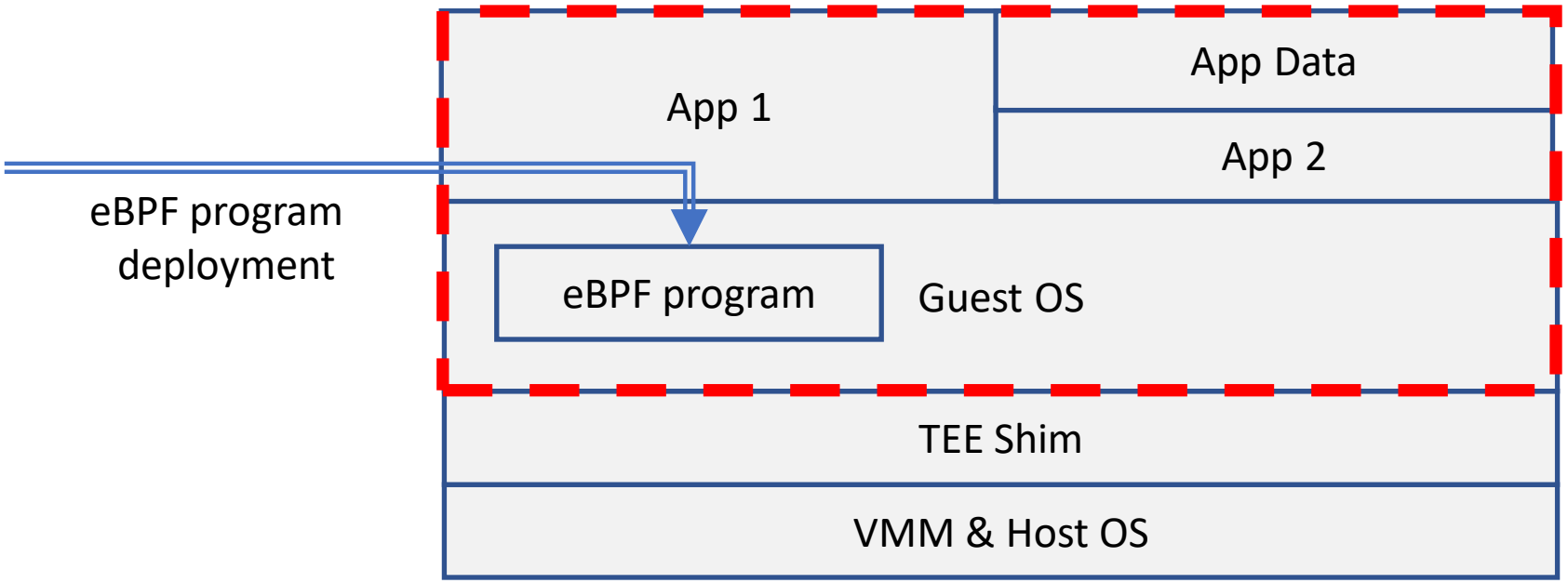
Example 1: CVM



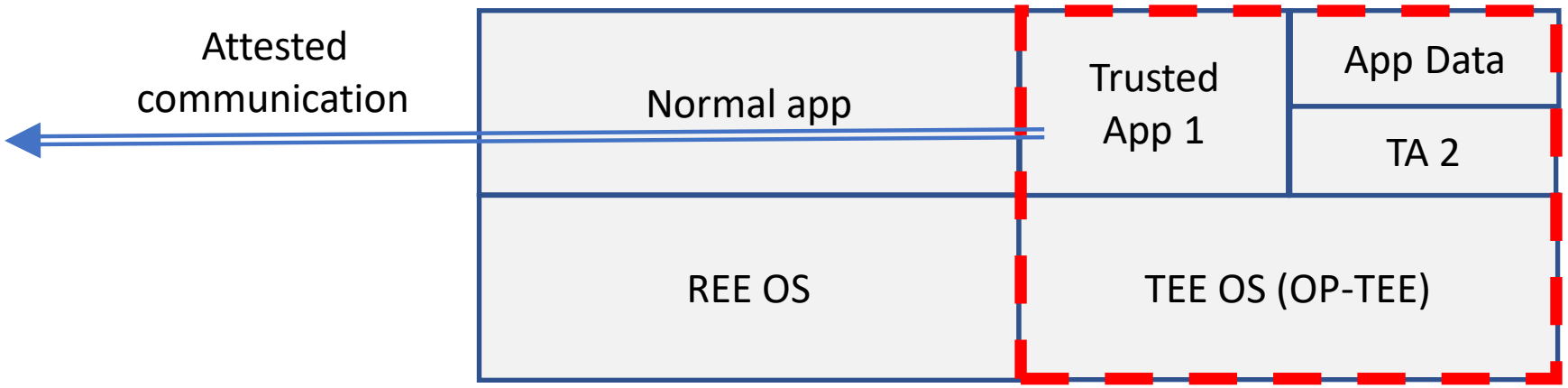
Example 1: VM in TDX or AMD SEV-SNP



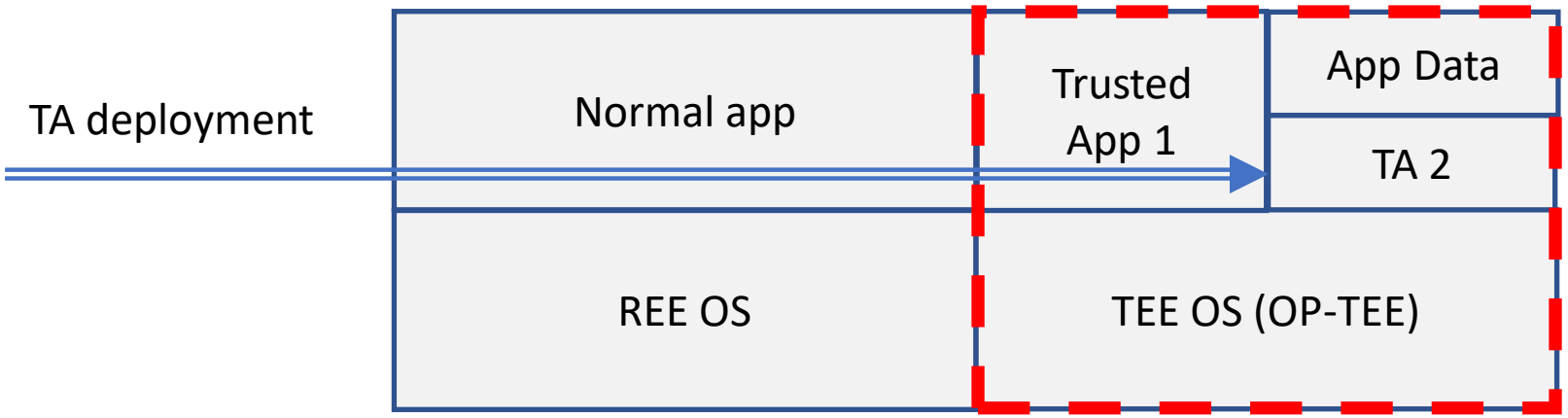
Example 1: VM in TDX or AMD SEV-SNP



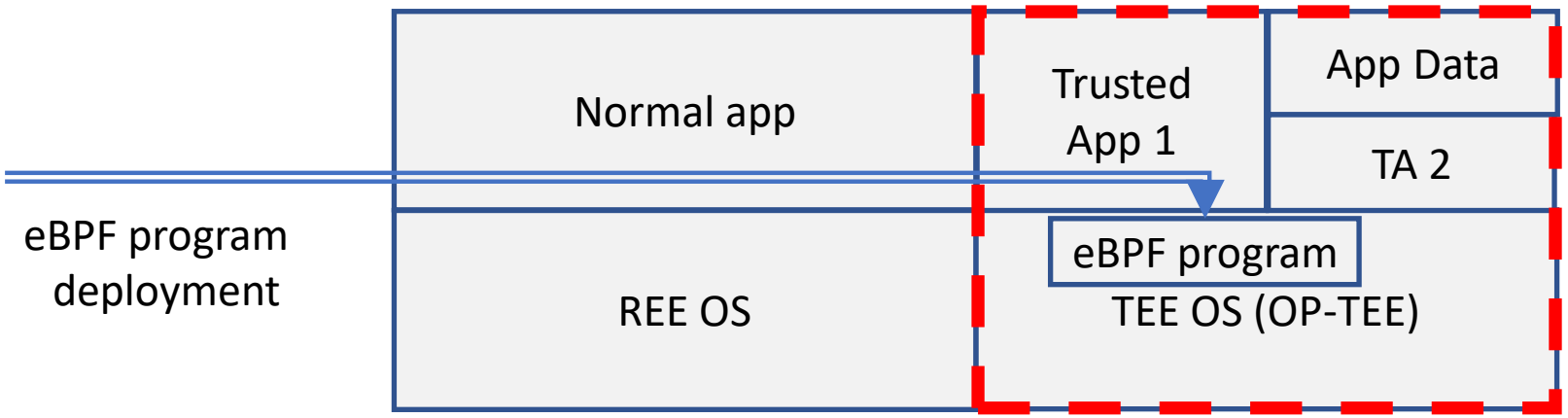
Example 2: OP-TEE



Example 2: OP-TEE



Example 2: OP-TEE



Attestation in eBPF + confidential computing

Many potential scenarios exist

1. Attestation in eBPF program deployment:
   A. deploy confidential eBPF program only to an attested TEE
   B. TEE only accepts eBPF programs from an attested source

2. eBPF extensions to attested communication:
   A. deploy code or data to an attested TEE (that has been extended with eBPF)
   B. only accept requests from an attested TEE (that has been extended with eBPF)
   Note: Since eBPF programs are usually deployed post-boot, boot-time attestation is insufficient

3. eBPF programs using attestation APIs:
   A. eBPF program that checks attestation in traffic
   B. eBPF program that checks attestation in APIs
   C. eBPF program as a verifier extension

Questions?