Trusted IO

- Trusted IO enables secure interactions between enclaves and external devices.
- Trusted path
  - Ensures confidentiality and integrity for the enclave's accesses to the device.
- Trusted device architecture
  - Protects enclave data on the device itself
  - Ensures device cannot leak sensitive data
  - Applies isolation principles to device resources
  - Particularly important for accelerators

Trusted Path Types

Logical Trusted Path

Uses access control mechanisms to allow/deny accesses based on origin or destination

- Access control filters for memory-mapped IO (MMIO)
- Trusted memory mappings

Cryptographic Trusted Path

Establishes a secure channel between enclave and device

- End-to-end encryption
- Authentication and attestation
- Can protect against physical attackers (Abus)

Some solutions combine both approaches, using different types for MMIO and DMA.

Trusted Device Architectures

For devices that process user data (e.g., accelerators), the device itself must protect data confidentiality and integrity.

Temporal Partitioning

Sharing device among multiple contexts over time with secure context switching

Spatio-temporal Partitioning

Multiple enclaves access device concurrently with hardware-enforced isolation

Cryptographic Protection

Applied to device-side memory resources similar to CPU DRAM protection

The isolation strategies used for CPU and memory can be applied to device resources as well.

Trusted IO Examples

GPU Protection

- Graviton: Hardware-enforced isolation
- Telekine: Cryptographic protection
- HIX: Logical path for MMIO, cryptographic for DMA
- ZeroKernel: Temporal isolation

FPGA Protection

- MeetGo: Secure remote applications
- ShEF: Shielded enclaves
- Trustore: Multi-tenant isolation

Secure Storage

Secure storage ensures that sensitive data persists across different enclave invocations and is only available to authorized entities.

Sealing

Process of encrypting data before storing it persistently

Unsealing

Process of decrypting data, accounting for enclave state

Binding Policies

Rules determining which enclaves can unseal previously sealed data

Only about a third of existing TEE solutions explicitly discuss sealing support, with most implementations resembling the original TPM-based approach.

Sealing Approaches

TPM-based Sealing

- Generate asymmetric key pair
- Encrypt data such that it can only be decrypted when system configuration matches
- Uses measurements in TPM's Platform Configuration Registers (PCRs)
- Examples: Flicker, SEA, IBM-PEF

Software TCB Sealing

- TCB exposes interface to create sealing keys
- No additional hardware required
- Examples: OP-TEE, Keystone, TIMBER-V, Sanctuary

Some TEEs like Intel SGX expose special CPU instructions in hardware to enable sealing, with different binding types (developer identity, enclave measurement).

Binding Policies for Sealed Data

Different TEEs implement various policies for determining which enclaves can unseal data:

Same Measurement

Only an enclave with identical measurement on the same platform can unseal the data

Same Developer

All enclaves signed by the same developer can unseal each other's data

Migration Support

Enclaves can come with a migration policy to transfer sealed data to a different host

The architecture must ensure that only the TCB and the owner enclave have access to the sealing keys.