HYPERLEDGER AVALON
(VIRTUAL MEETUP WITH HYPERLEDGER AVALON AND CHAINLINK)

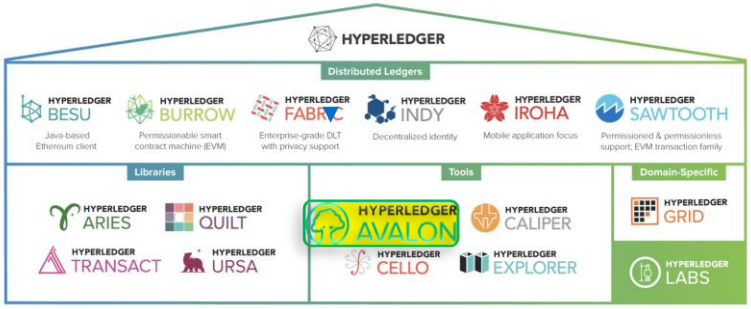
Eugene (Yevgeniy) Yarmosh
08/06/2020

Agenda

- Hyperledger Avalon at Glance
- Confidential Compute and Trusted Execution Environments (TEE) Intro
- Background and Usages
- Challenges of Bridging Confidential Compute and Blockchains
- Architecture

Extending Computational Trust beyond Blockchain

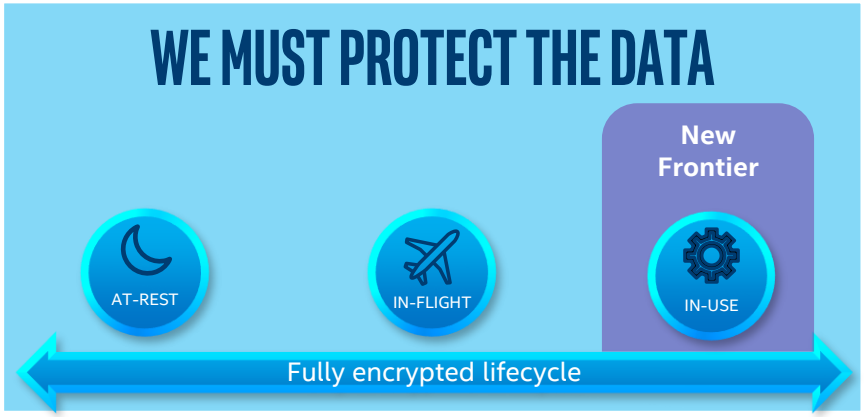
- Hyperledger Avalon
  - Improves blockchain scalability and confidentiality
  - Utilizes HW based Trusted Execution Environments
  - Running side-by-side with blockchain
  - Blockchain neutral
- Standard based reference implementation
  - Implements EEA (Enterprise Ethereum Alliance) Off-Chain Trusted Compute (TC) Specification
  - Top level Hyperledger project sponsored by 16 companies and organizations
- Links
  - TC spec: Off-Chain-TC-Spec-Link
  - Github: https://github.com/hyperledger/avalon
  - Project: Proposal and Announcement



- Alibaba Cloud
- Baidu
- Banco Santander
- BGI
- Chainlink
- Consensys
- EEA
- Espeo
- IBM
- Intel
- iExec
- Kaleido
- Microsoft
- Monax
- Oracle
- WiPro

Confidential Compute and Trusted Execution Environments (TEE) Intro

Confidential Computing Protects Data IN-USE

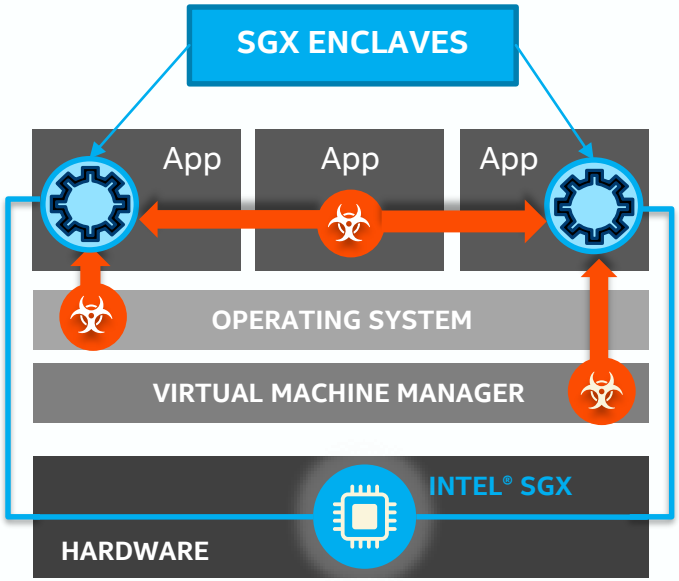


Protecting data in use by performing computation within
hardware-based Trusted Execution Environments (TEE),
especially, when it takes place in a platform or environment, we do not directly control

TEE provides secure enclaves, memory encryption and attestation

TEE - Trusted Execution Environment

Intel® Software Guard Extensions, aka SGX as an example



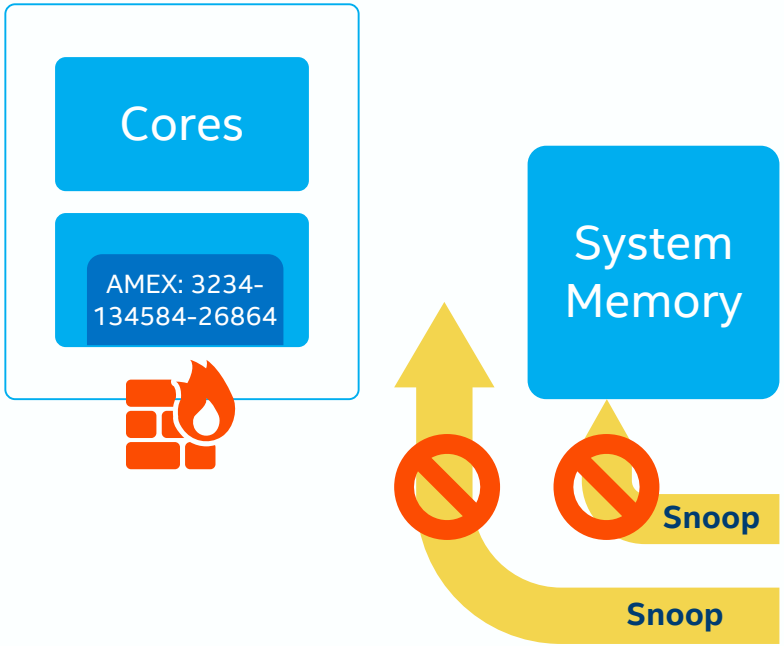
Protects against:

- Malicious insiders
- External attackers
- 3rd-Parties without consent

Intel® SGX Based TEE Attributes:

- A trusted enclave can be created by user mode app at any time
- Enclave isolates code and data from app, OS, and VMM
- Built-in extendable attestation mechanism (verifiable by a remote party)

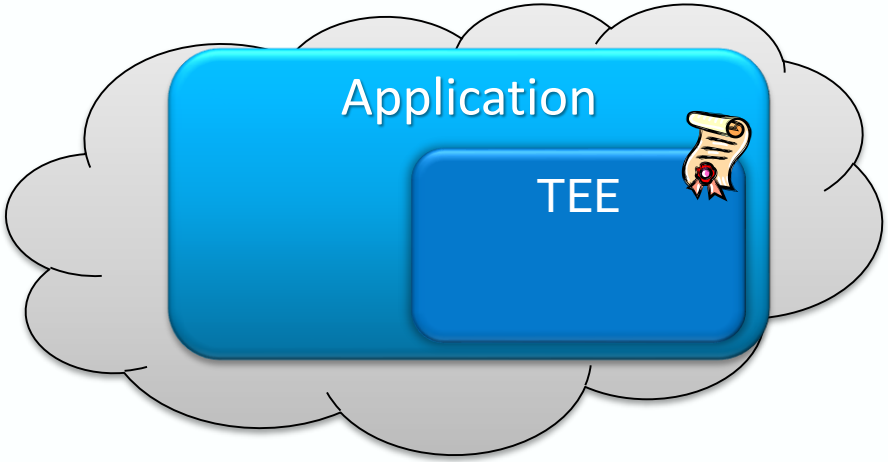
MEMORY Protection Outside CPU



- Security perimeter is the CPU package boundary
- Data and code unencrypted inside CPU package
- Data and code outside CPU package is encrypted and integrity checked
- Increases likelihood that external memory reads and bus snoops see only encrypted data

Attestation: Code Integrity & Authenticity

- **You deploy (signed) code on the remote platform**
- HW signs an attestation of the enclave identity and TCB level
- Deploy secrets and trust execution results

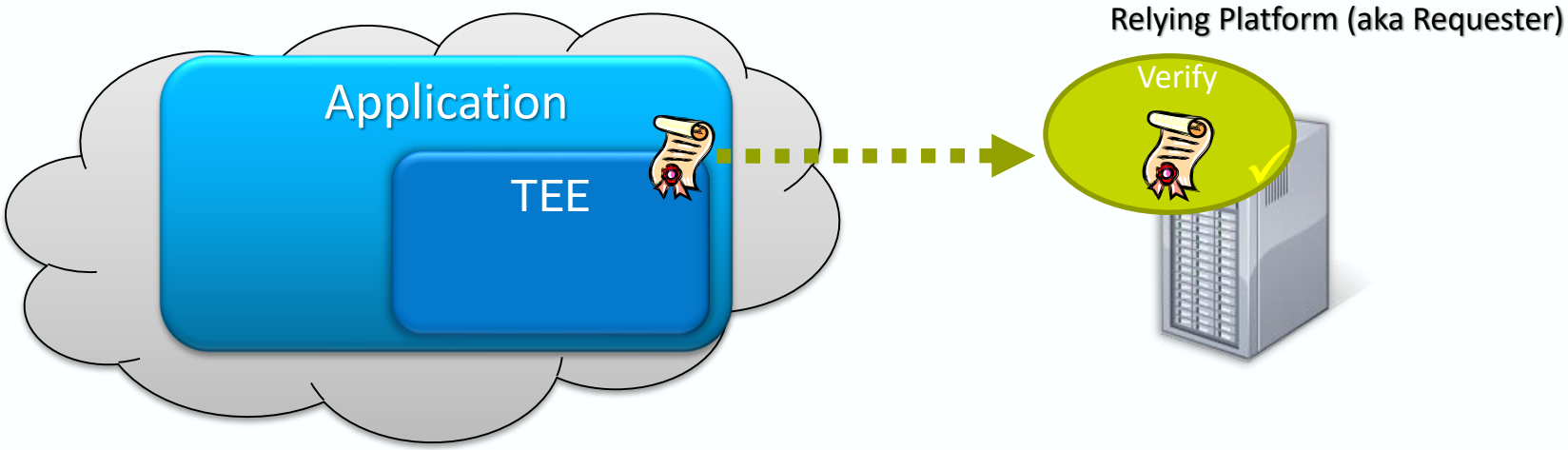


Relying Platform (aka Requester)

Enclave memory is measured against your (signed) code

Attestation: Code Integrity & Authenticity

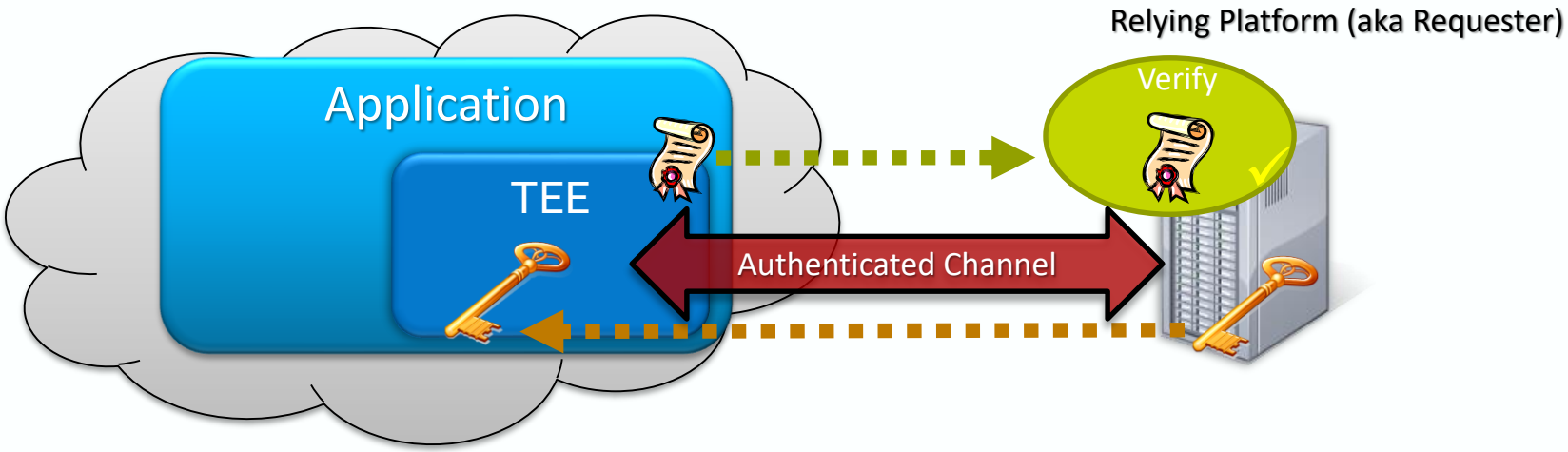
- You deploy (signed) code on the remote platform
- **HW signs an attestation of the enclave identity and TCB level**
- Deploy secrets and trust execution results



This is the right app executing in the right platform

Attestation: Code Integrity & Authenticity

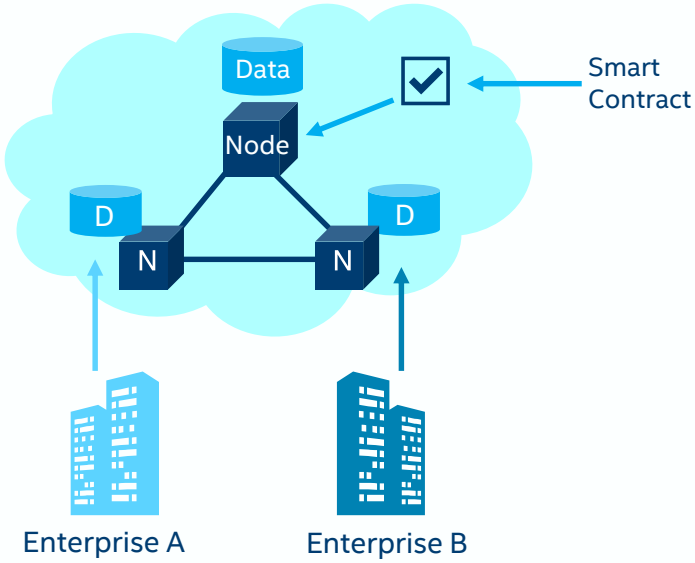
- You deploy (signed) code on the remote platform
- HW signs an attestation of the enclave identity and TCB level
- **Deploy secrets and trust execution results**



Protecting keys, code (e.g. scripts, models), input and output data

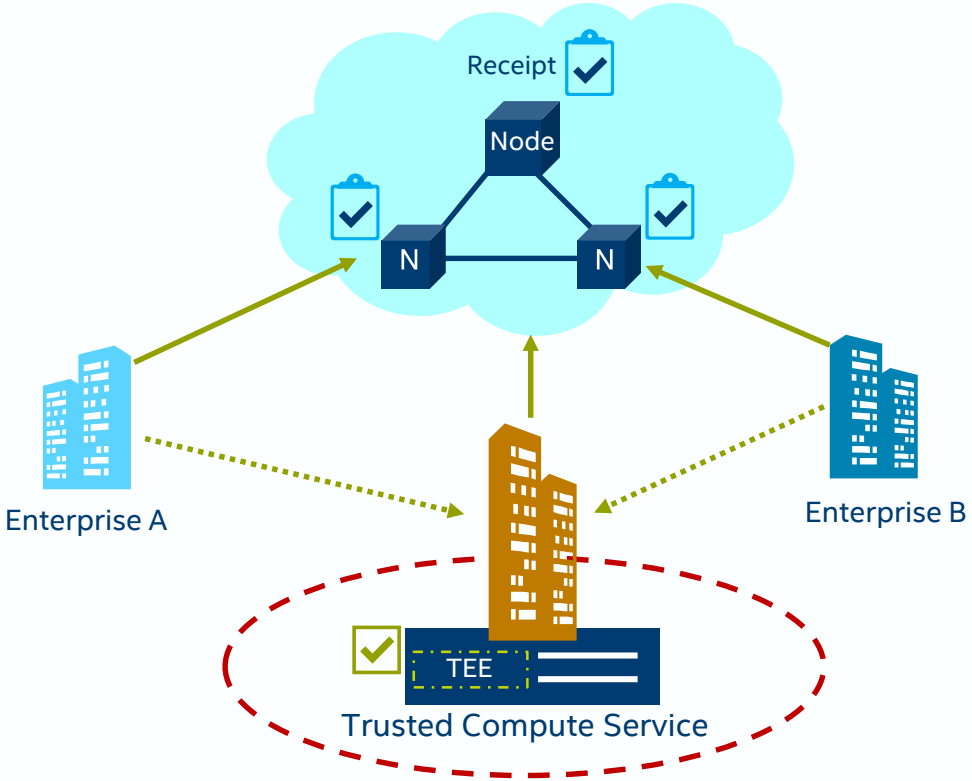
Background and Usages

On-chain execution of smart contracts



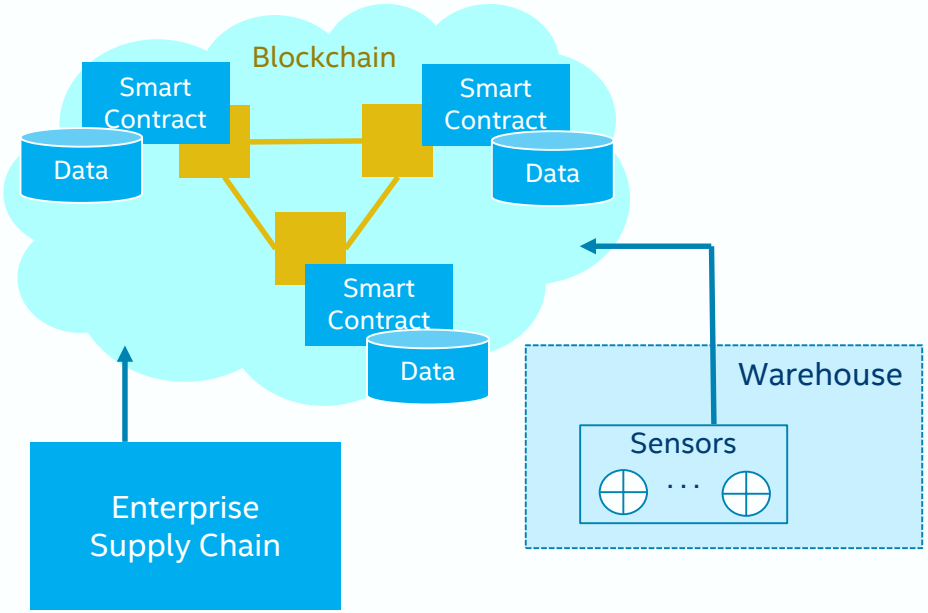
- Enterprise A and Enterprise B agree to execute a smart contract with terms and policies specific to their transaction.
- The smart contract is executed on the blockchain
- The transaction data is visible to all participants on the chain and is recorded on each blockchain node.
- Communication is facilitated through the blockchain.

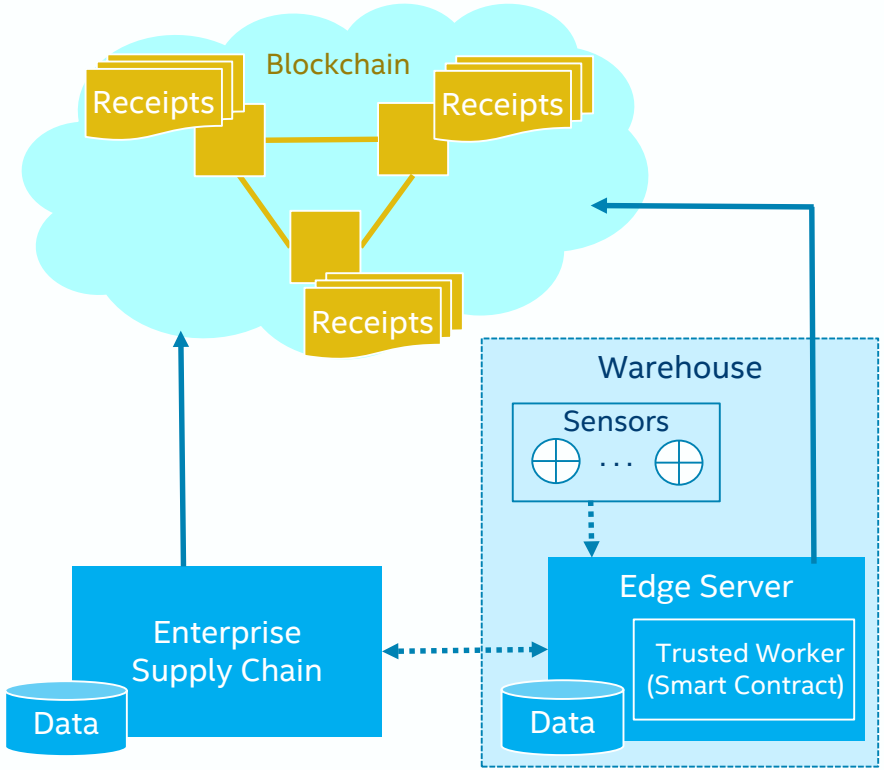
Off-chain execution of smart contracts



- Enterprises A and B execute a smart contract off-chain by a Trusted Compute Worker
- The TC Worker (code, platform) can be attested and the attestation can be verified by other parties
- The attested TC Worker (aka black box) is the only who can see full transaction details and it enforces transaction policy, confidentiality and integrity
- The blockchain stores transaction receipts – cryptographic representations of the transactions
- Communication between the TCS and enterprises can be direct via JRPC or through the blockchain

IoT with Edge Sensors

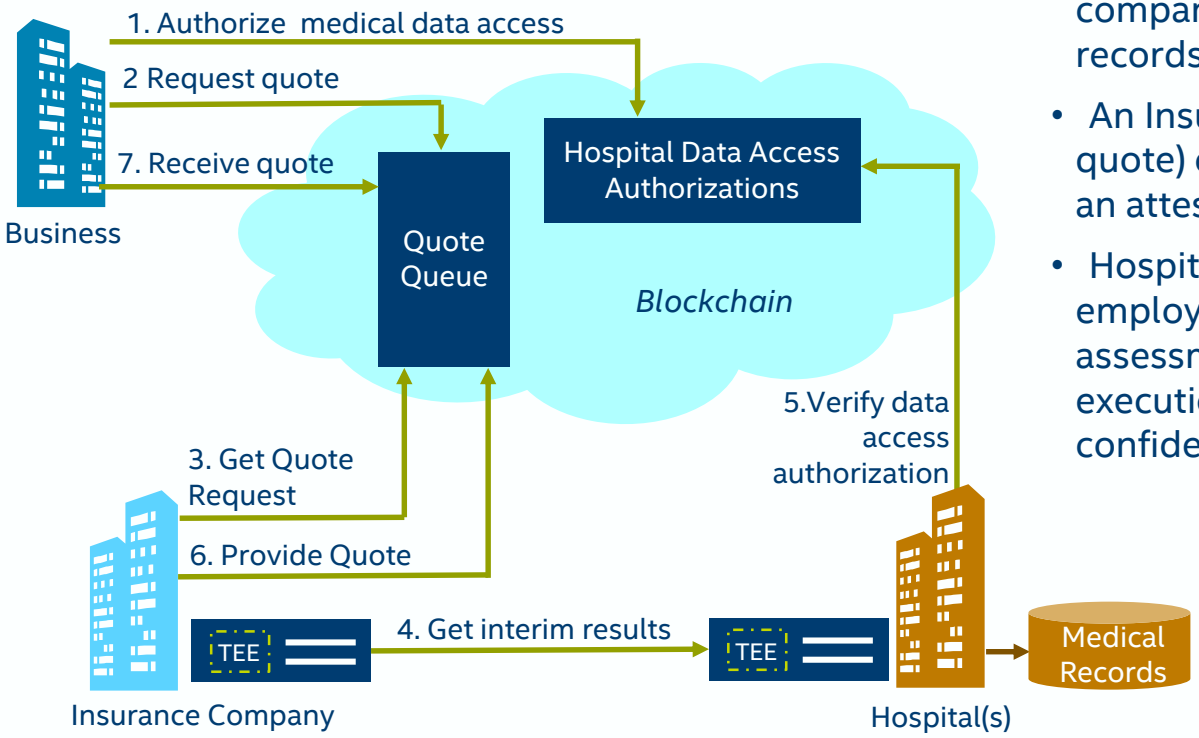




- *The smart contract is executed on the blockchain*
- *Execution is replicated by each blockchain node*
- *Sensor data are universally visible, stored on the blockchain*

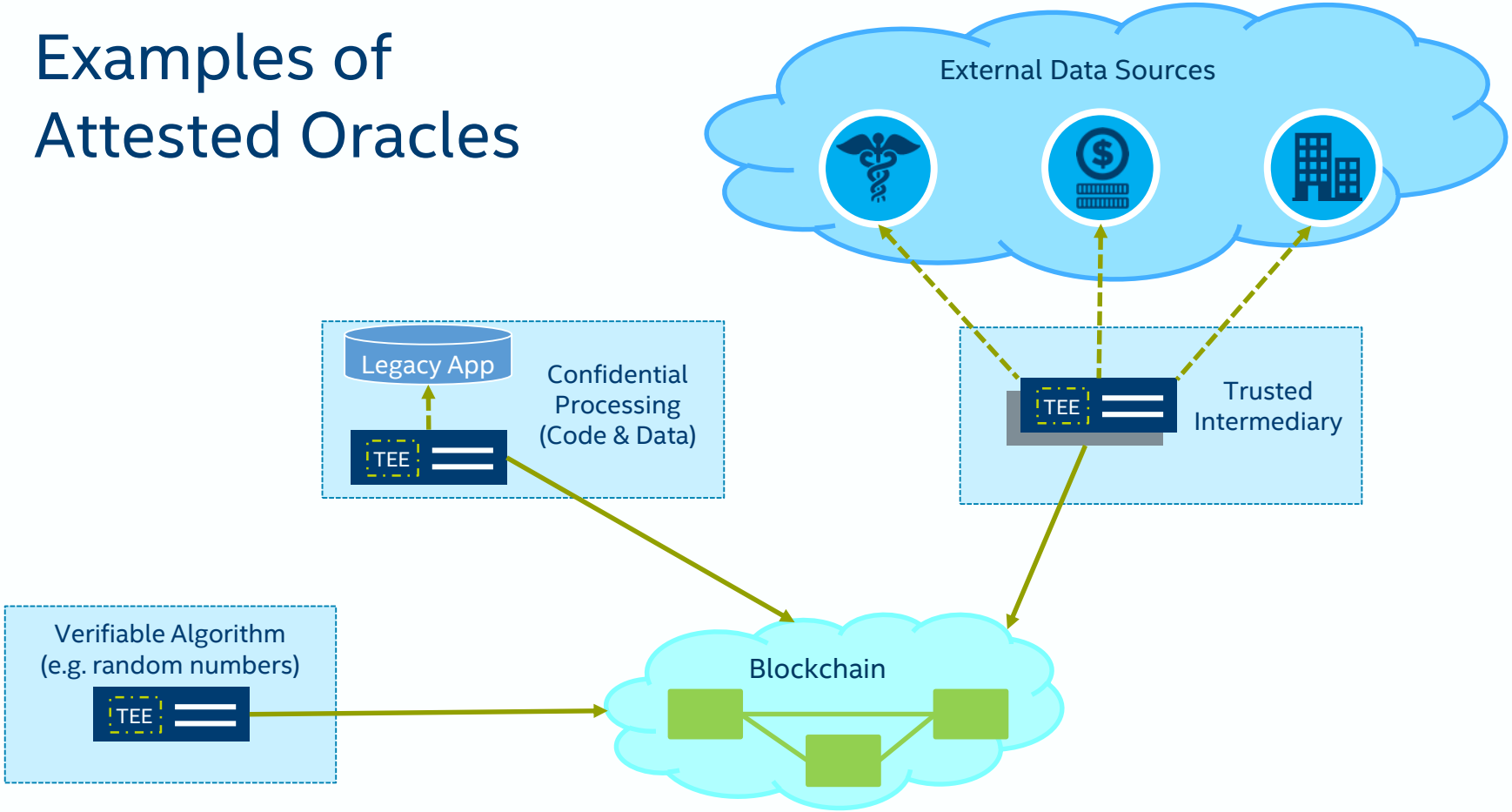
- *Smart contract execution is off-chain; the blockchain is an arbiter*
- *Smart contract is executed securely within a trusted worker*
- *Sensor data stored according to the contract policies*

Confidential Compute Example

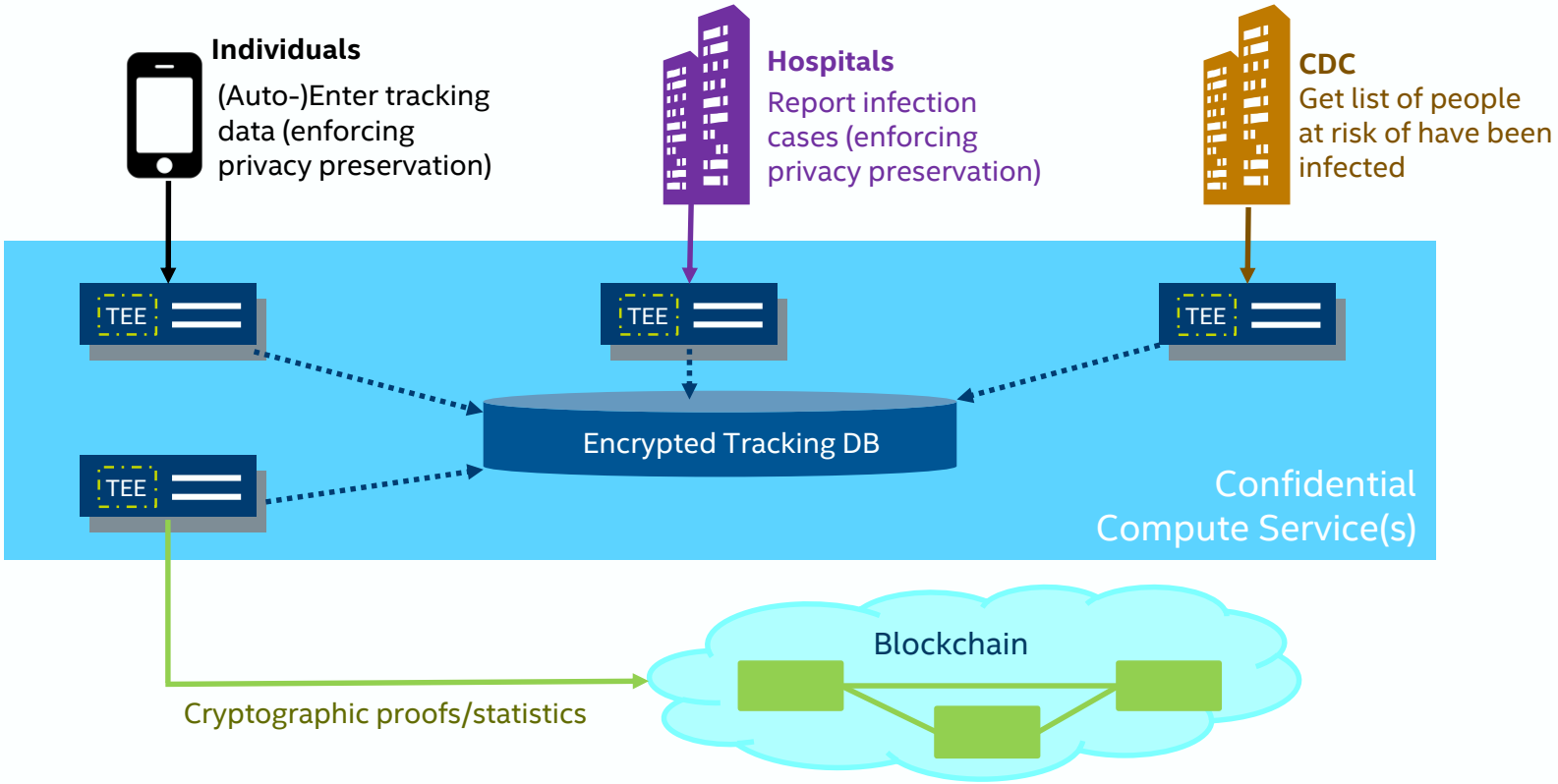


- A Business wants to get a group medical insurance quote from the insurance company, authorizes access to medical records of its employees
- An Insurance Company (to determine a quote) calculates a heart disease risk factor in an attested trusted execution environment
- Hospitals maintain medical records of the employees and provide them for the assessment, but only in an attested trusted execution environment (that enforces confidentiality)

Examples of Attested Oracles

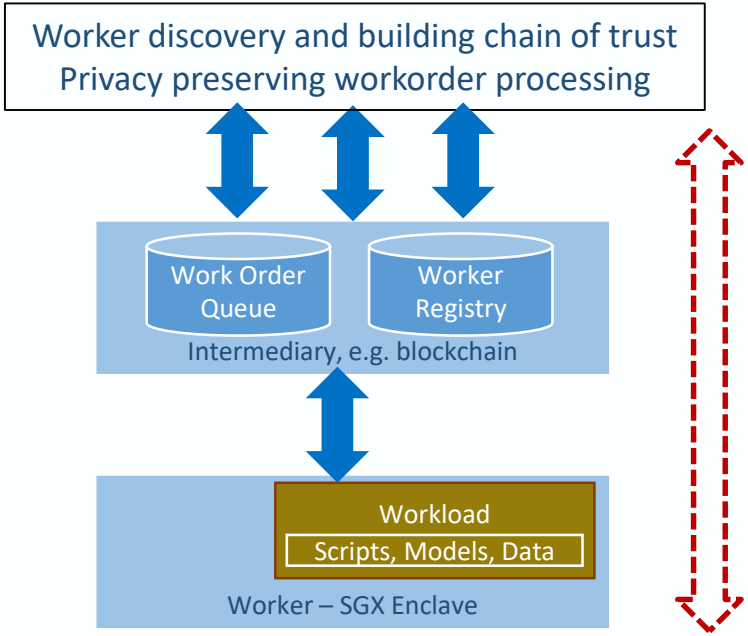


Example of Privacy Preserved Covid19 Tracking



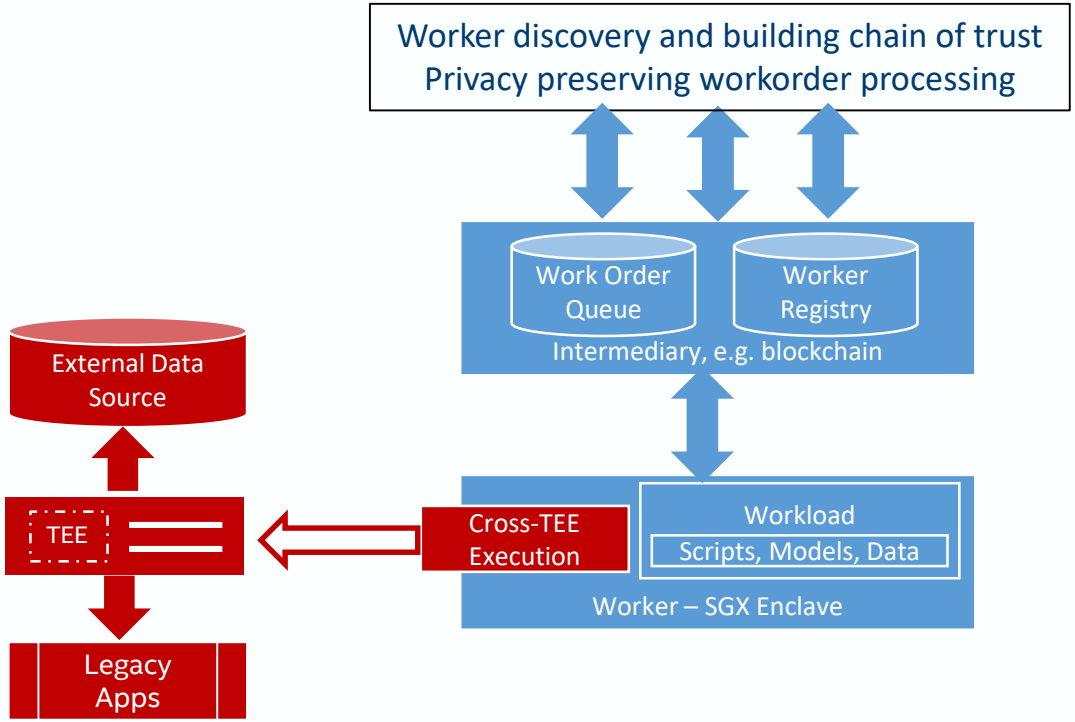
Challenges of Bridging Confidential Compute and Blockchains

Asynchronous Model



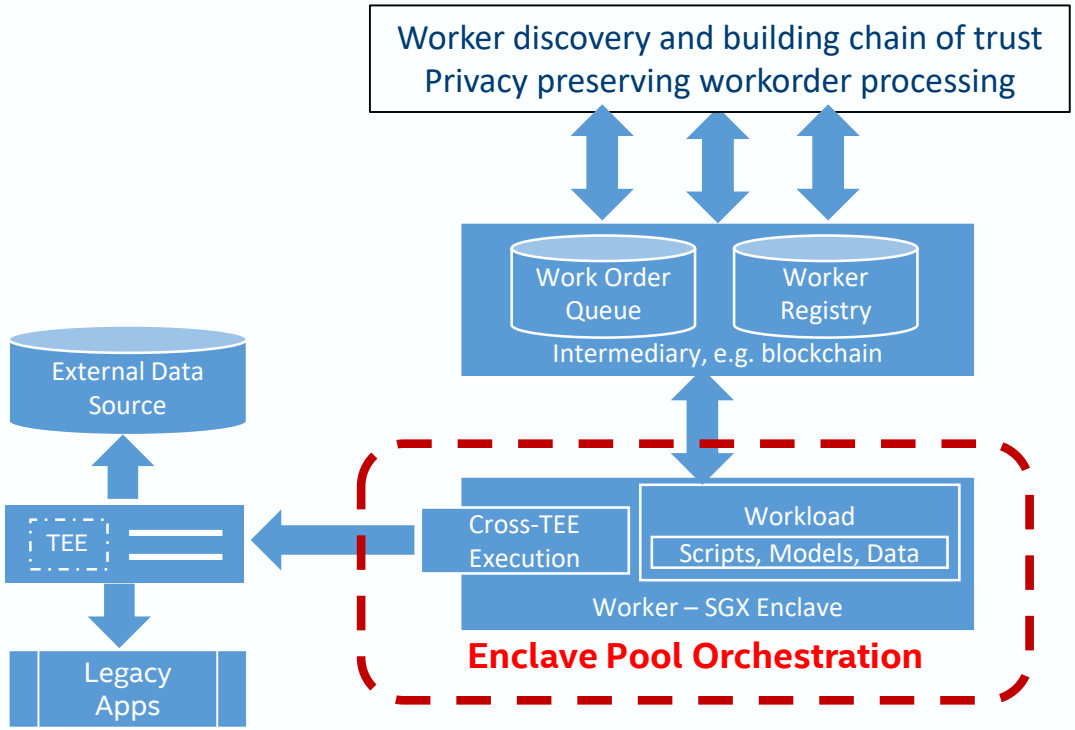
- **Establishes asynchronous compute model**
  Standard based, critical for decentralized usages and micro-services (aka FaaS)

Cross-TEE Execution



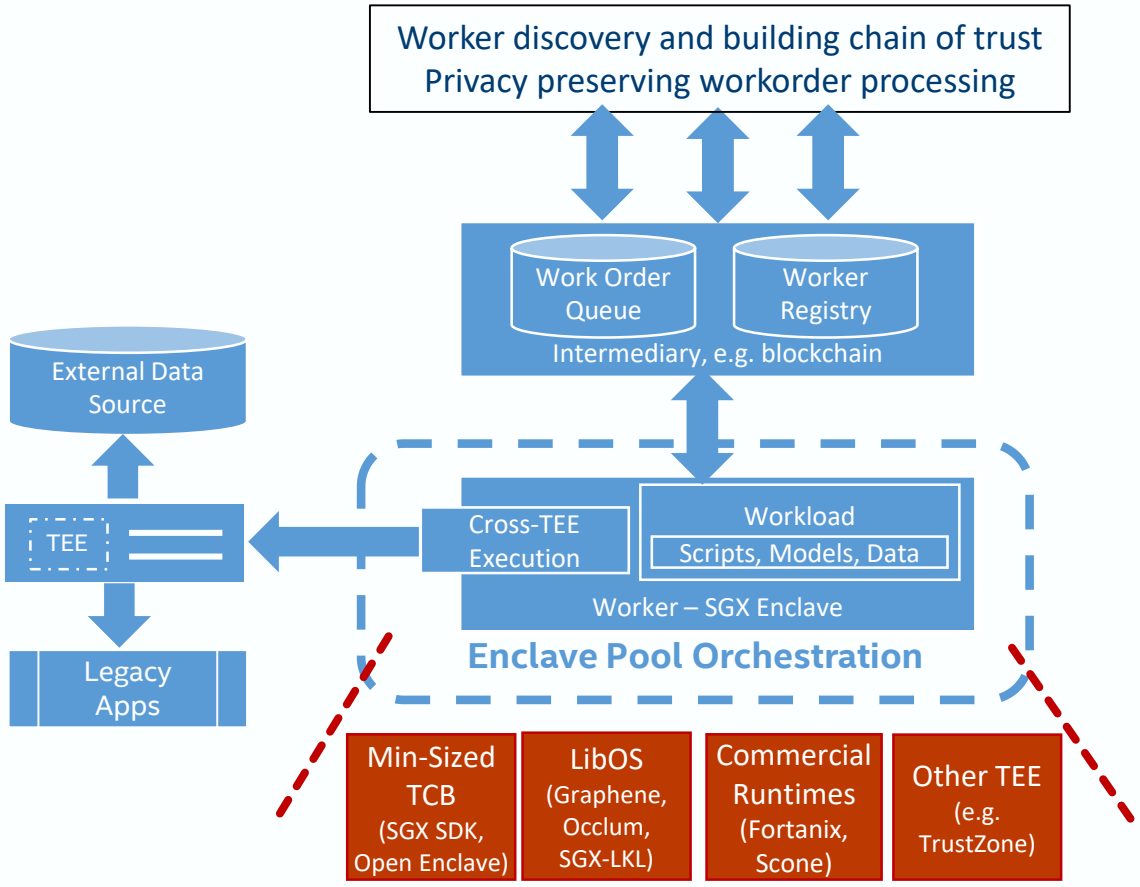
- **Establishes asynchronous compute model**
  Standard based, critical for decentralized usages and micro-services (aka FaaS)
- **Facilitates Cross-TEE Execution**
  Enables new usages (CFL) and improves integrity of links to legacy apps, sensors, data

Scalability



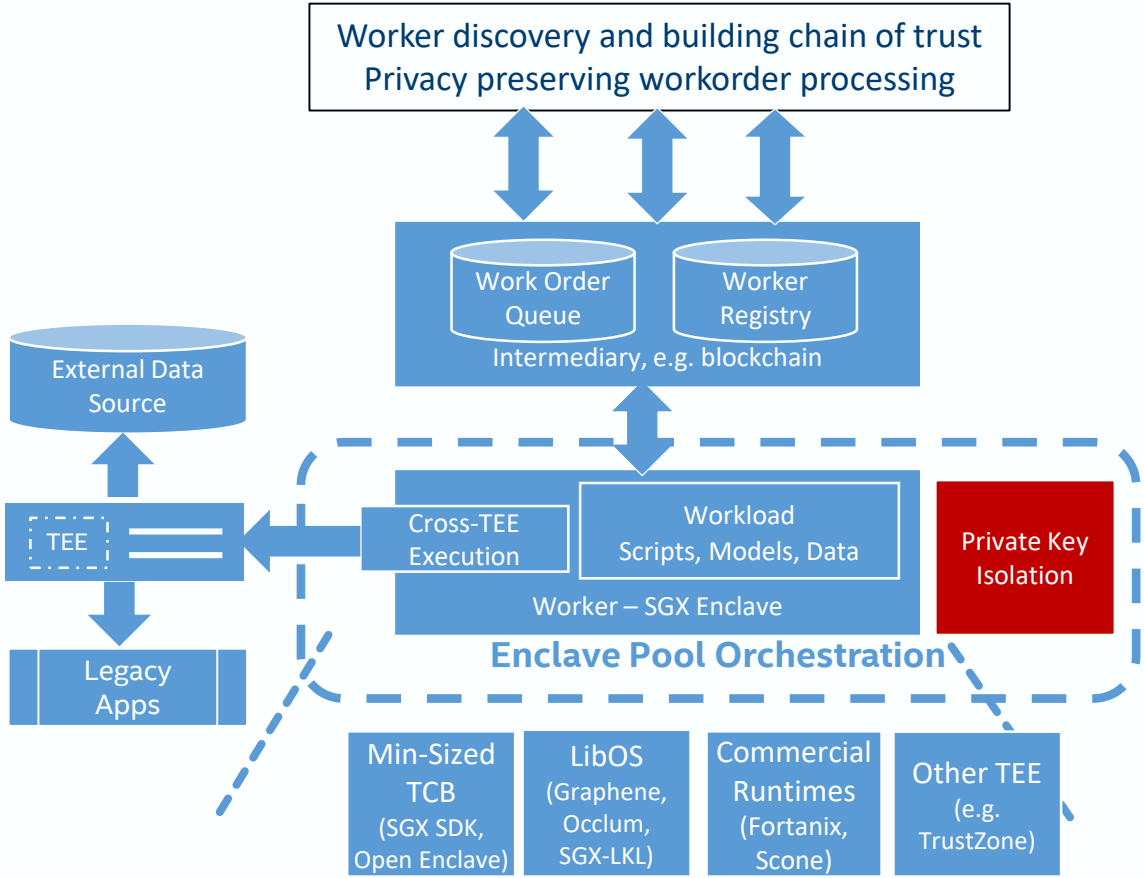
- **Establishes asynchronous compute model**
  Standard based, critical for decentralized usages and micro-services (aka FaaS)
- **Facilitates Cross-TEE Execution**
  Enables new usages (CFL) and improves integrity of links to legacy apps, sensors, data
- **Enables scalability & multitenancy**
  Addressing SGX bond to a specific system. Works with existing orchestrators, K8S, OpenNESS

Integration of Various TEE Runtimes



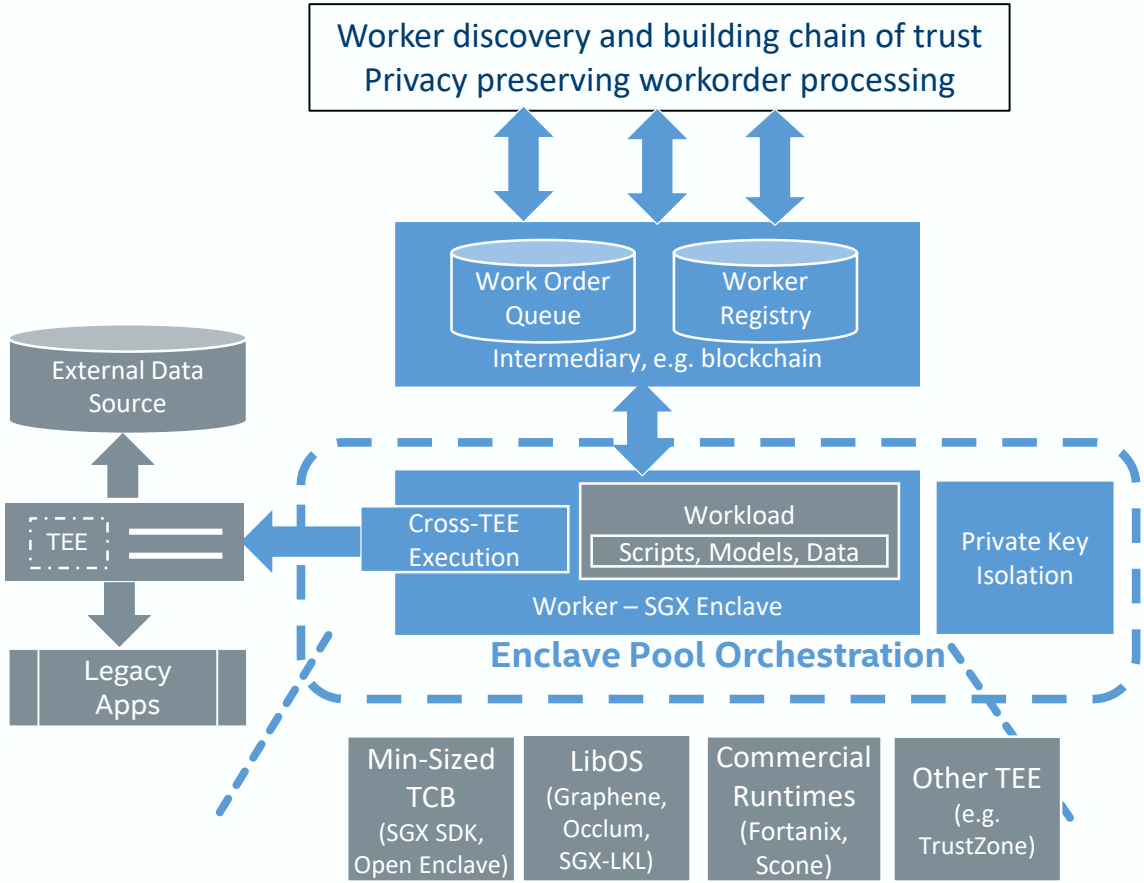
- **Establishes asynchronous compute model**
  Standard based, critical for decentralized usages and micro-services (aka FaaS)
- **Facilitates Cross-TEE Execution**
  Enables new usages (CFL) and improves integrity of links to legacy apps, sensors, data
- **Enables scalability & multitenancy**
  Addressing SGX bond to a specific system. Works with existing orchestrators, K8S, OpenNESS
- **Supports variety of TEE options**
  Mixing apps with different TCBs (SGX SDK, Open Enclave, LibOS runtimes) and different TEEs

Mitigating Potential Application Vulnerabilities



- **Establishes asynchronous compute model**
  Standard based, critical for decentralized usages and micro-services (aka FaaS)
- **Facilitates Cross-TEE Execution**
  Enables new usages (CFL) and improves integrity of links to legacy apps, sensors, data
- **Enables scalability & multitenancy**
  Addressing SGX bond to a specific system. Works with existing orchestrators, K8S, OpenNESS
- **Supports variety of TEE options**
  Mixing apps with different TCBs (SGX SDK, Open Enclave, LibOS runtimes) and different TEEs
- **Reduces impact of vulnerabilities**
  By isolating key management from the work order execution

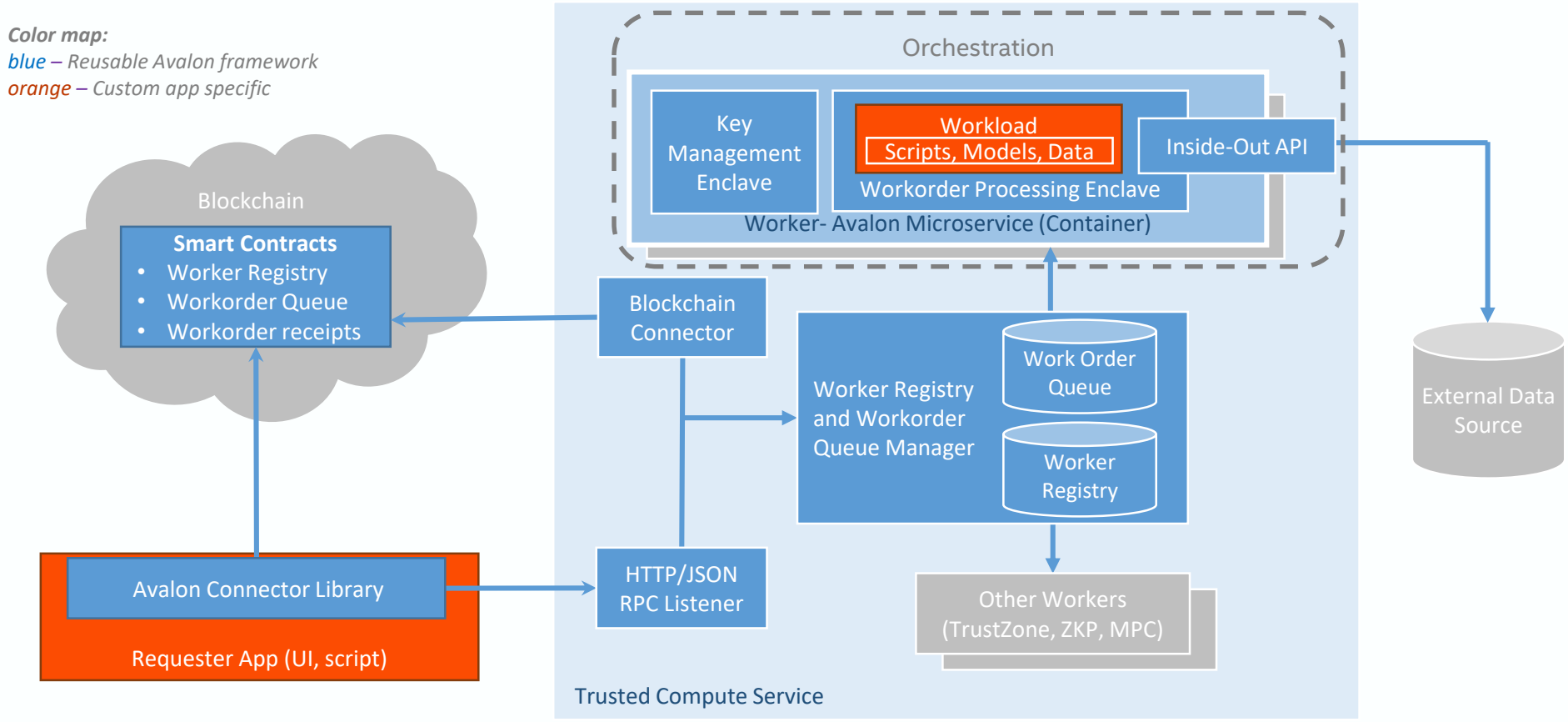
Avalon Objectives



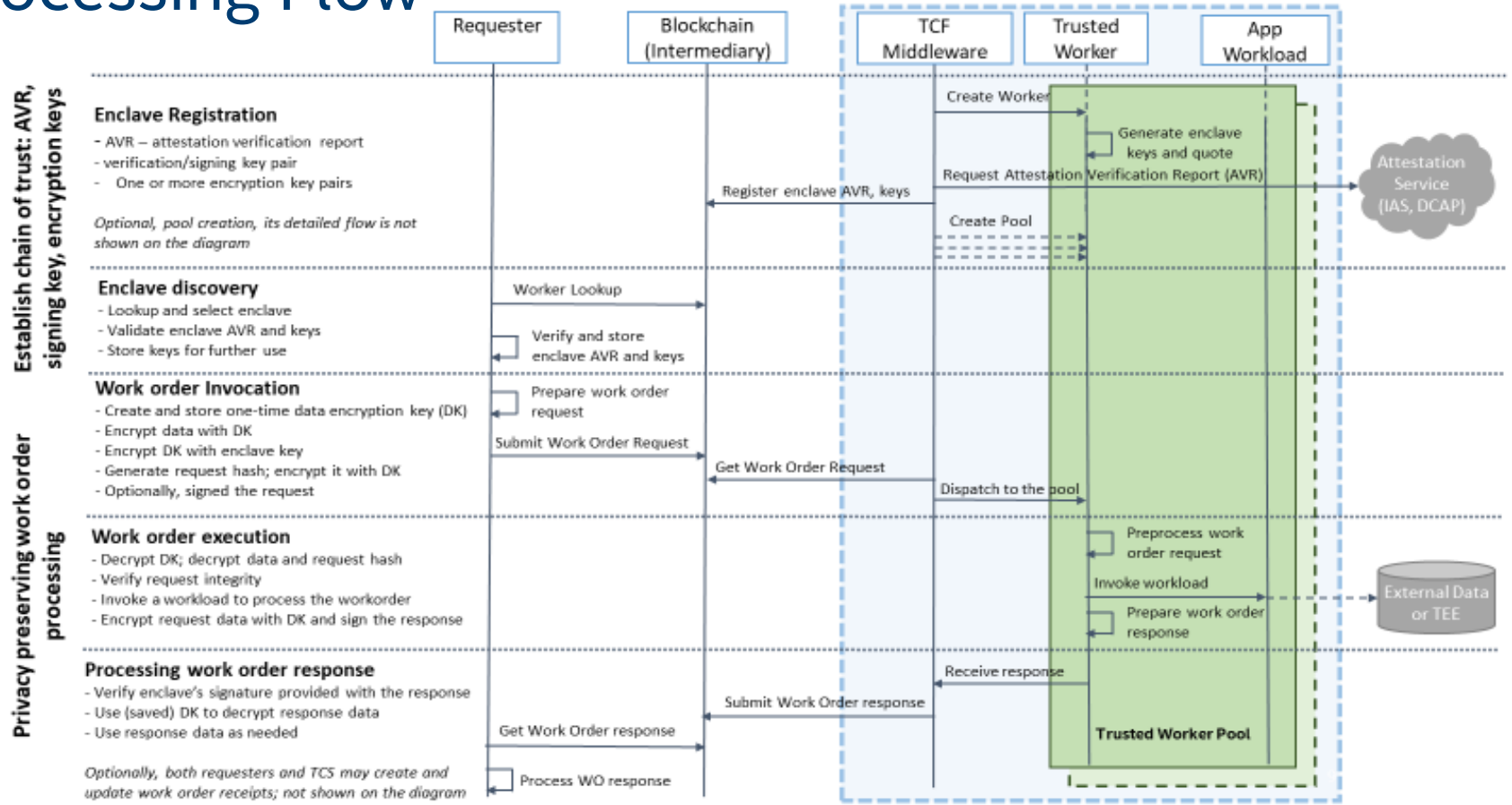
- **Establishes asynchronous compute model**
  Standard based, critical for decentralized usages and micro-services (aka FaaS)
- **Facilitates Cross-TEE Execution**
  Enables new usages (CFL) and improves integrity of links to legacy apps, sensors, data
- **Enables scalability & multitenancy**
  Addressing SGX bond to a specific system. Works with existing orchestrators, K8S, OpenNESS
- **Supports variety of TEE options**
  Mixing apps with different TCBs (SGX SDK, Open Enclave, LibOS runtimes) and different TEEs
- **Reduces impact of vulnerabilities**
  By isolating key management from the work order execution

Architecture

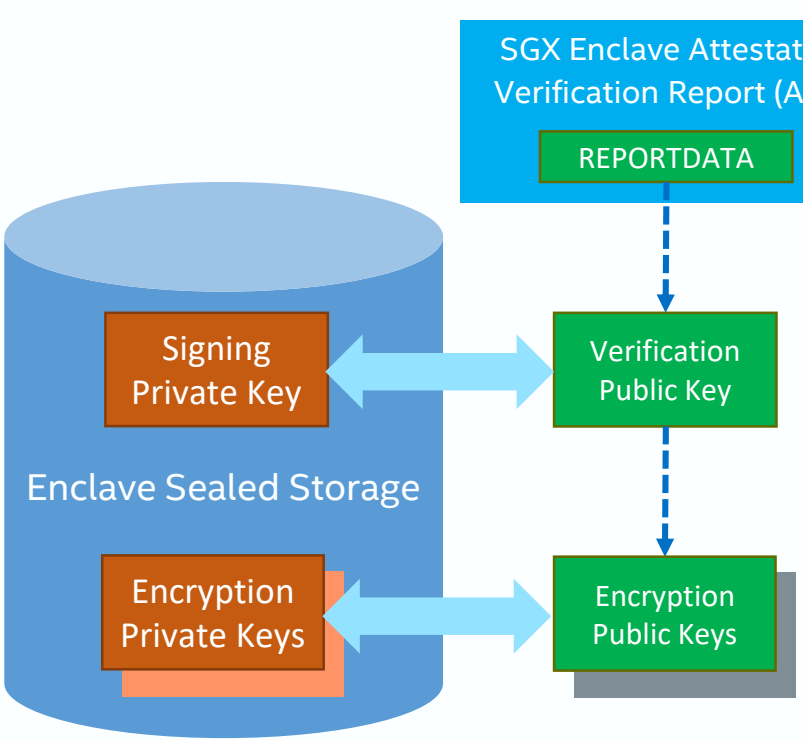
Hyperledger Avalon High Level Architecture



Processing Flow



Chain of Trust

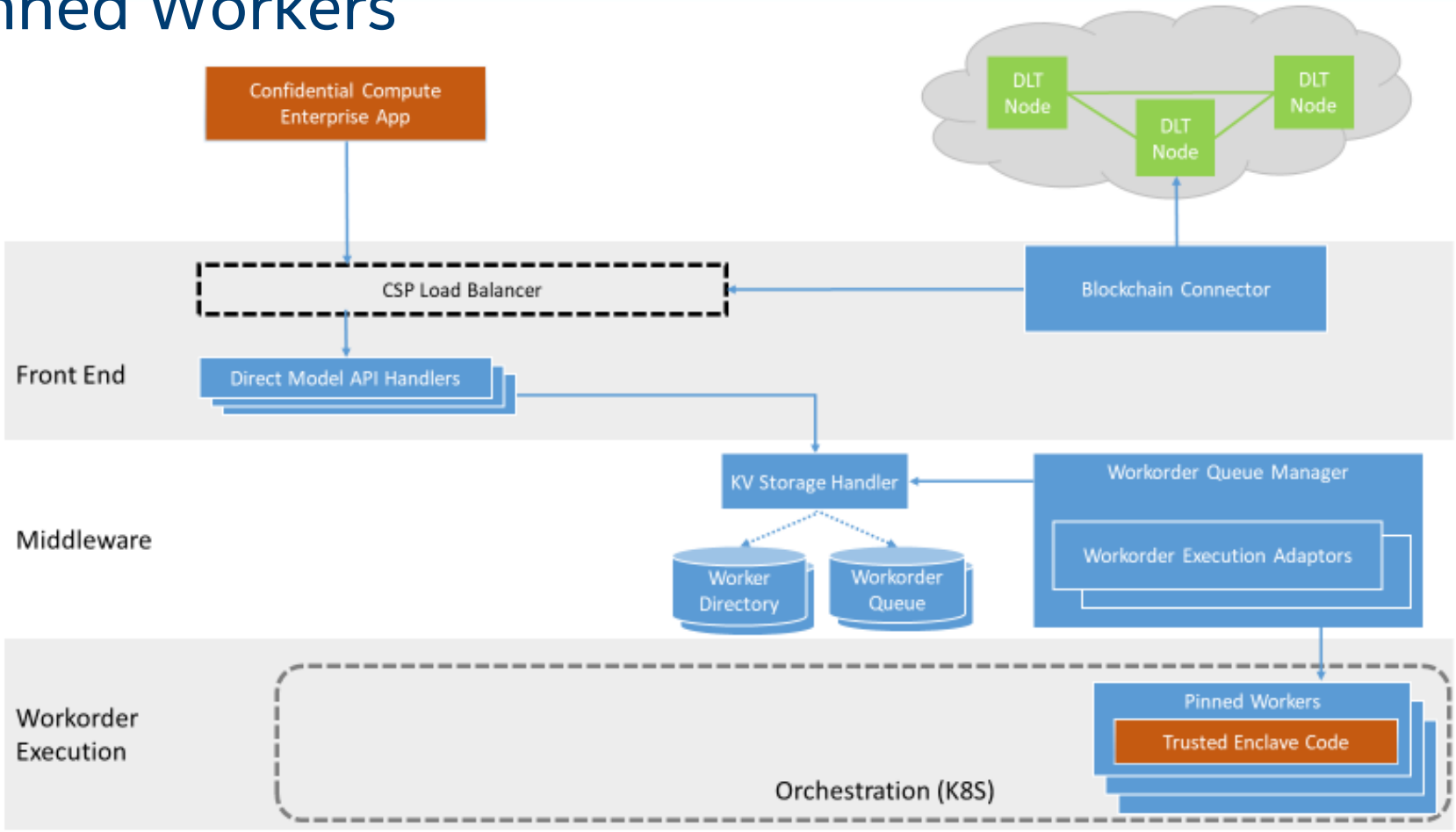


- Enclave Generates Asymmetric Signing/Verification Key Pair
  - One signing key pair per the enclave life time
  - Used to sign encryption keys and workorder results
  - Verification Public Key is published on blockchain
  - Verification Public Key Hash in the AVR's REPORTDATA
  - Private key is stored in the enclave sealed storage
- Enclave Generates Asymmetric Encryption Key Pairs
  - One or more encryption keys pairs per enclave
  - Used to encrypt one-time symmetric transaction key
  - There can be reusable or one-time keys
  - Encryption key life-time is defined by app policy
  - Public key is published on blockchain
  - Public key is signed by the enclave signing key
  - Reusable private key is stored in the enclave sealed storage
  - One-time private key is stored in the enclave memory (not shown)

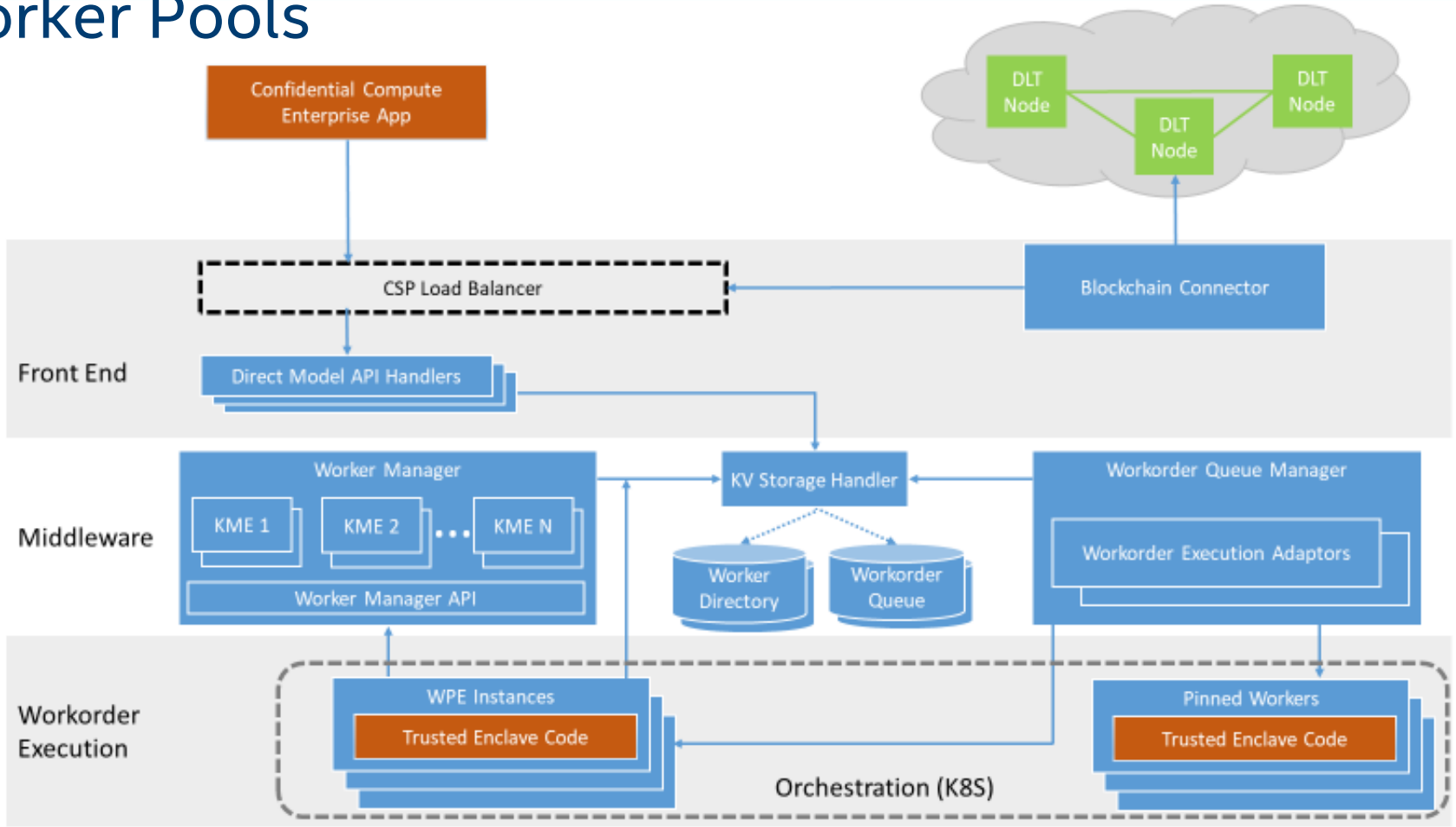
Work Order Confidentiality and Integrity

- Requester
  - Generates one time symmetric encryption AES-GCM-256 key (SEK)
  - Encrypts work order request data with SEK
  - Calculates SHA256 hash of the request and encrypts it with SEK
  - Encrypts SEK with enclave's public (RSA-OAEP) encryption key
  - Optionally, signs the SHA256 hash using its verification key
- Worker (trusted code inside of SGX enclave)
  - Decrypts one time symmetric encryption AES-GCM-256 key (SEK)
  - Decrypts work order request data
  - Calculates SHA256 hash of the request
  - Decrypts SHA256 hash provided in the request and compares it with the calculated value above
  - If the signature of request's SHA256 hash is provided, verifies it using requester's key
  - Processes work order
  - Encrypts work order response data with SEK
  - Calculates SHA256 hash of the response
  - Signs the hash value with enclave's private signing key

Pinned Workers



Worker Pools



Thank you!

Get involved with Hyperledger Avalon

- GUTHUB: https://github.com/hyperledger/avalon
- Tutorial: https://github.com/hyperledger/avalon/tree/master/docs/workload-tutorial
- Docs, links: https://github.com/hyperledger/avalon/tree/master/docs