RSAConference2018
San Francisco | April 16 – 20 | Moscone Center

SESSION ID: EXP-W12

THE RISE OF CONFIDENTIAL COMPUTING

Mark Russinovich

Azure CTO
Microsoft

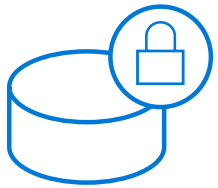
Cloud Data Threats

Customer cloud data concerns:

- Malicious privileged admins or insiders
- Hackers exploiting bugs in the Hypervisor/OS of cloud fabric
- Third parties accessing it without customer consent

Data Protection

At rest



Encrypt inactive data when stored in blob storage, database, etc.

Examples:

- Azure Storage Service Encryption for Data at Rest
- SQL Server Transparent Database Encryption (TDE)

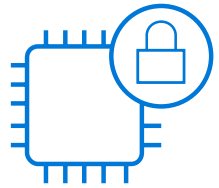
In transit

Encrypt data that is flowing between untrusted public or private networks

Examples:

- HTTPS
- TLS

In use



Protect/Encrypt data that is in use during computation

Examples include:

- Trusted Execution Environments
- Homomorphic encryption

Trusted Execution Environments (TEEs)

Protected container:

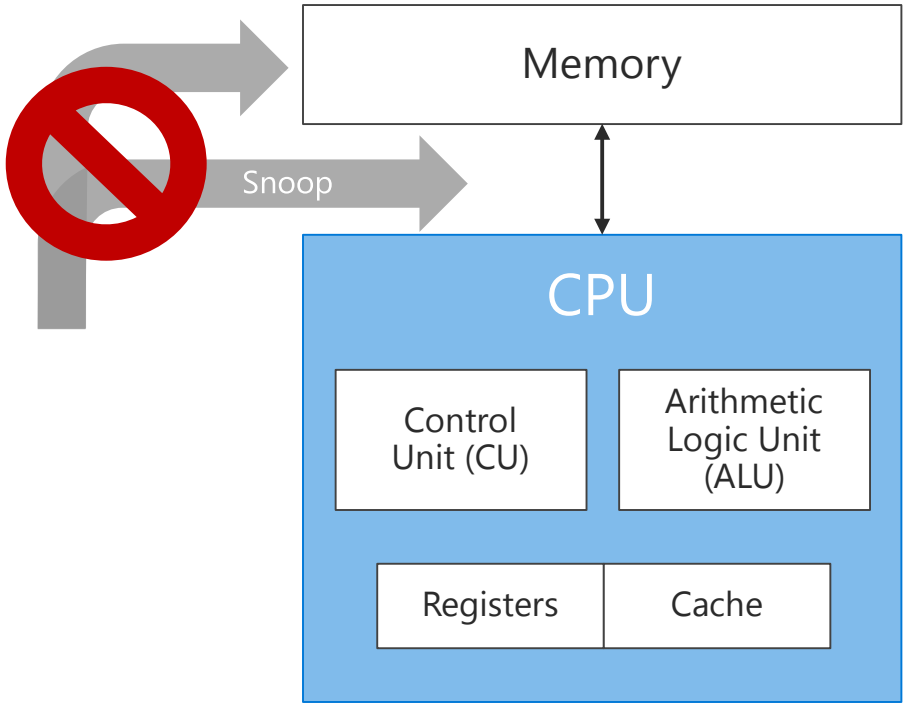
- Isolated portion of processor & memory
- Code & data cannot be viewed or modified from outside

Supports attestation: proving of identity both locally and remotely

Supports sealing: persisting secrets

Examples:

- Intel SGX
- Virtualization Based Security (VBS) aka Virtual Secure Mode



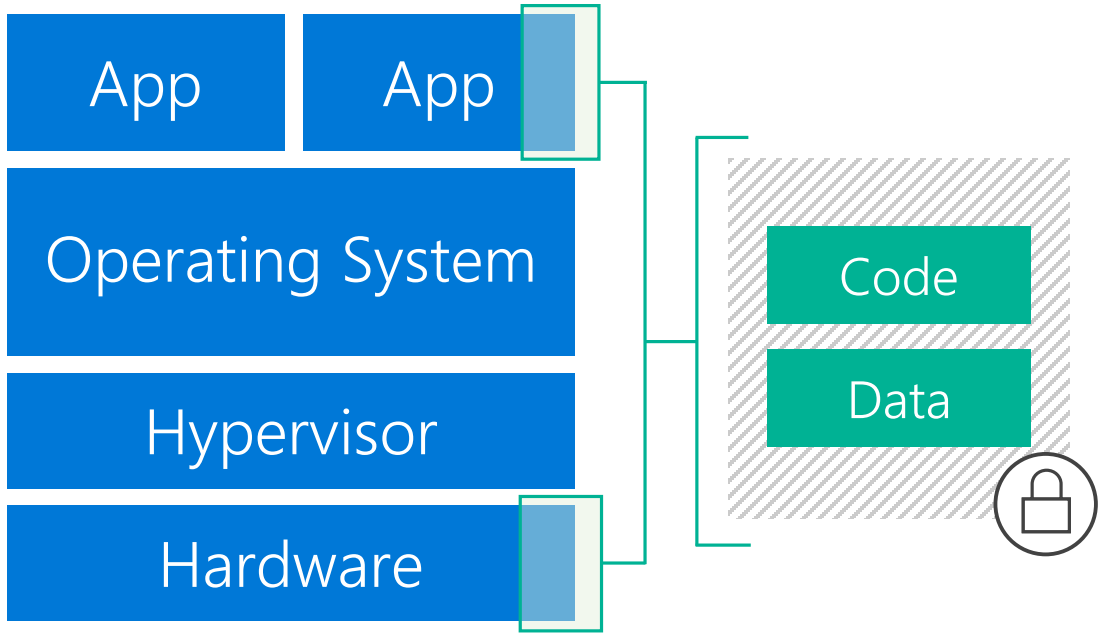
Hardware-based TEE

Intel SGX (Software Guard Extensions)

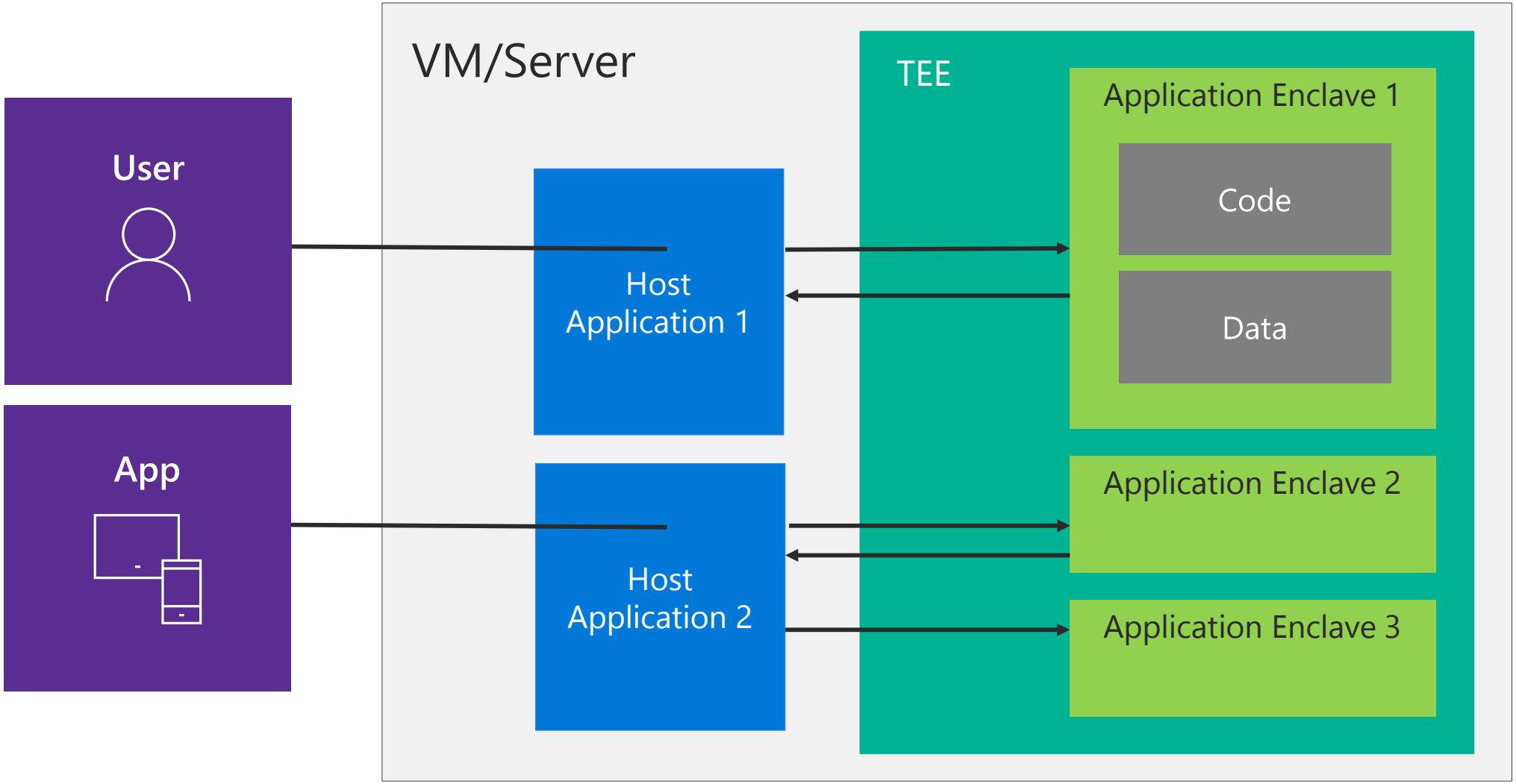
SGX goal: Minimize hardware attack surface

Instructions to set aside private regions ("protected containers") of code and data

Data is only in clear within protected memory and CPU



TEE application architecture

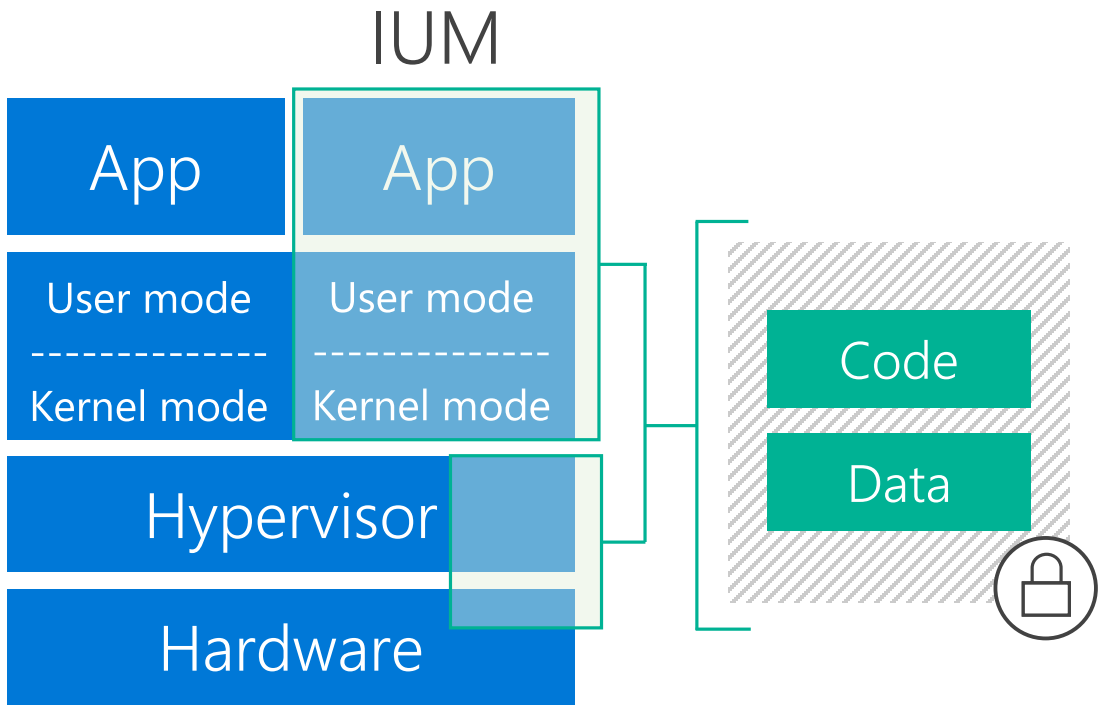


Hyper-V Virtualization Based Security (VBS)*

Virtual trust layers (VTLs) identify access levels

VTL 1 defines Isolated User Mode (IUM)

VBS prevents the OS, applications in VTL 0 and device drivers from accessing IUM (VTL 1)



*AKA Virtual Secure Mode (VSM)

TEEs compared to other secure hardware

	TPMs	HSMs	TEE (Intel SGX)
Hardware	Separate physical chip; embedded into motherboard	Separate external device	Built into CPU
Operations	Secure cryptographic operations	Secure cryptographic operations*	Secure "container" in which to run arbitrary code
Attestation	Yes	Yes	Yes
Sealing	Yes	Yes	Yes
Use case	Local system integrity • Validate properties at boot • Host measurement • Platform device authentication	Key management system • Generate & manage keys • Key exchange • Encryption	Generalized compute
Example application	BitLocker Disk Encryption	KeyVault	SQL Always Encrypted

*Some permit arbitrary code, but not optimized for general purpose compute

Common TEE application patterns

Protect data confidentiality and integrity on remote machine

Protect sensitive algorithmic IP (e.g. financial trade algorithms)

Code access security, including remote clients

Create a trusted network of nodes among a set of untrusted parties

Centrally combine different data sources for a better algorithmic outcome without loss of confidentiality

Protect communication with other secure device endpoints (e.g. local peer to remote HSM)

Secure licensing and DRM

Azure Confidential Computing

Confidential cloud

Data is fully in the control of the customer regardless of whether in rest, transit, or use even though the infrastructure is not

The cloud platform provider is outside the **trusted compute base**

Code running in cloud is protected and verified **by the customer**

Azure and confidential computing

Working with silicon partners to enable Confidential Computing

Building software to deploy, manage, and develop secure TEE applications on Azure

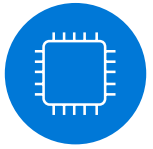
Designing and developing services to support attestation in the cloud

Enabling confidential PaaS and SaaS services

The ACC development environment

Universal
Generalize enclave application model to minimize hardware/ software specific concepts

Pluggable
Componentization to support desired runtimes and crypto libraries



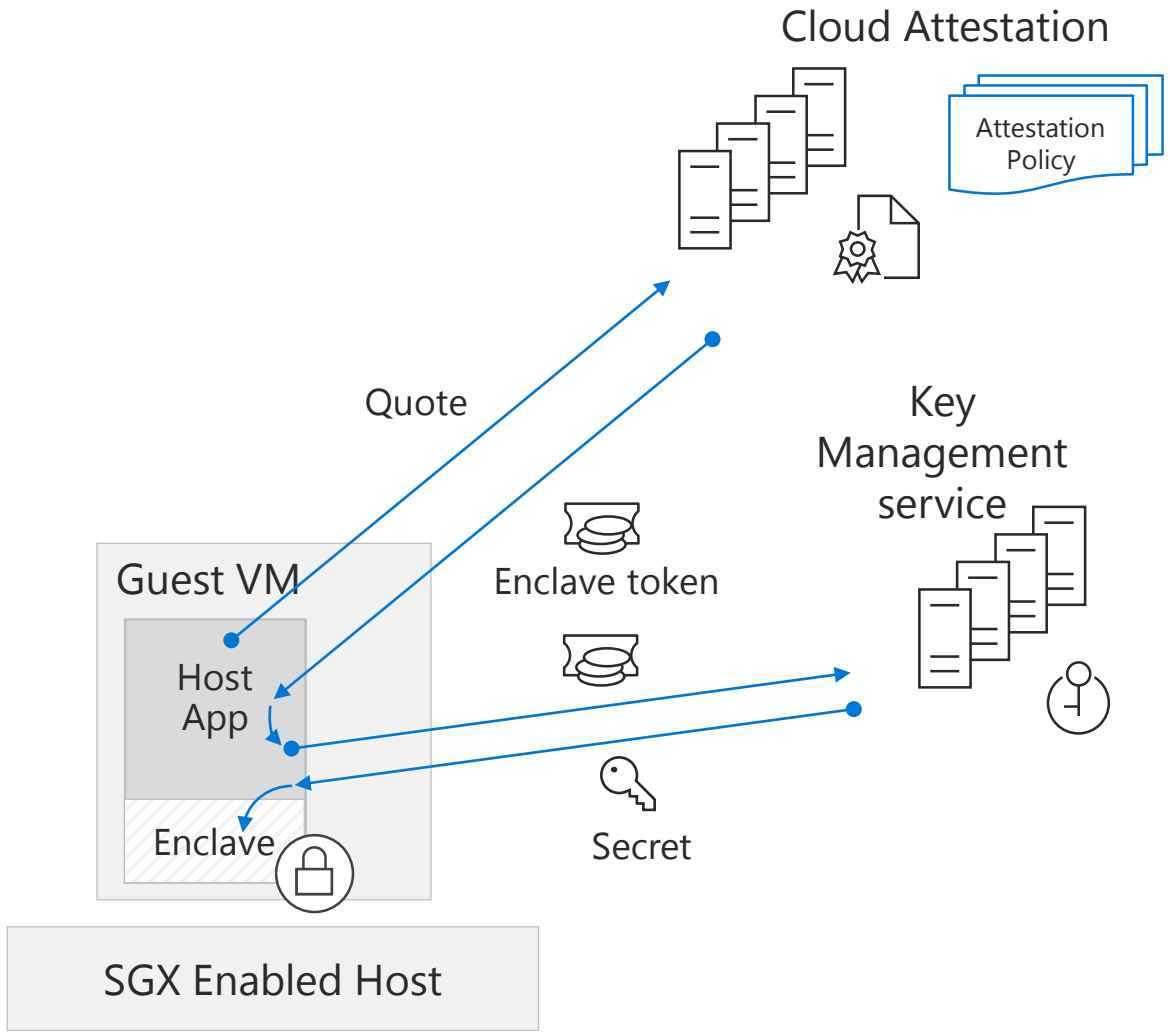
Standardized
Remove hardware vendor specific signing and verification requirements

Multi-platform
Design with all software platforms, Windows and Linux, in mind

Compatible
Easier enablement of redistributable applications

Open
Open source and a standard for secure enclave-based application

Universal cloud attestation

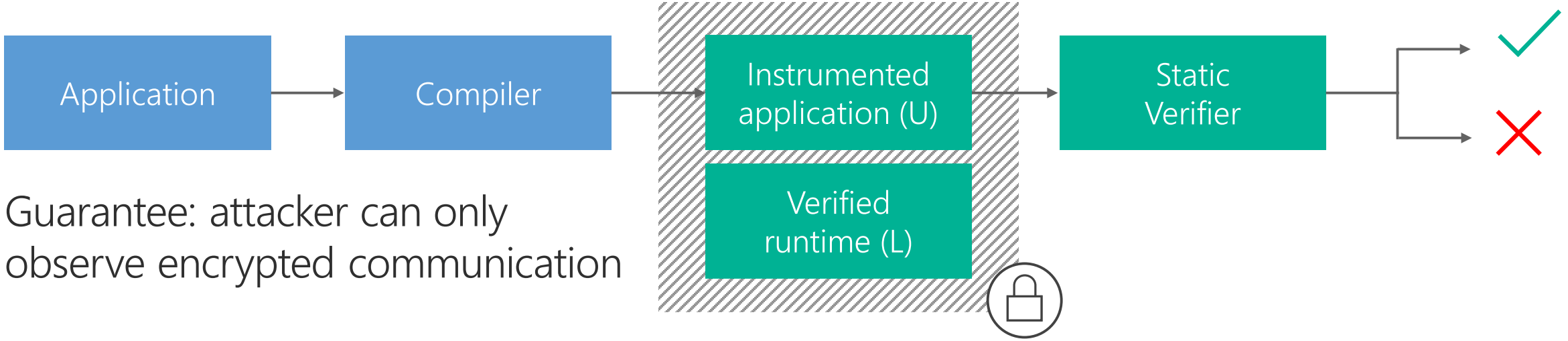


1. Quote provide proofs:
   - Code runs in genuine SGX enclave
   - Enclave version and owner is as expected
   - Arbitrary enclave supplied data is as expected
2. Attestation service attests to hardware, rooted in Intel chain of trust, and issues token
3. Cloud service (e.g. AKV) is presented with token with certs chained to CPU
4. Token is used to release secrets to the application enclave

Preventing direct information leaks

Problem: code in enclaves may unintentionally write secrets out

Solution: use a compiler that instruments memory accesses & verify that instrumented binary does not leak secrets



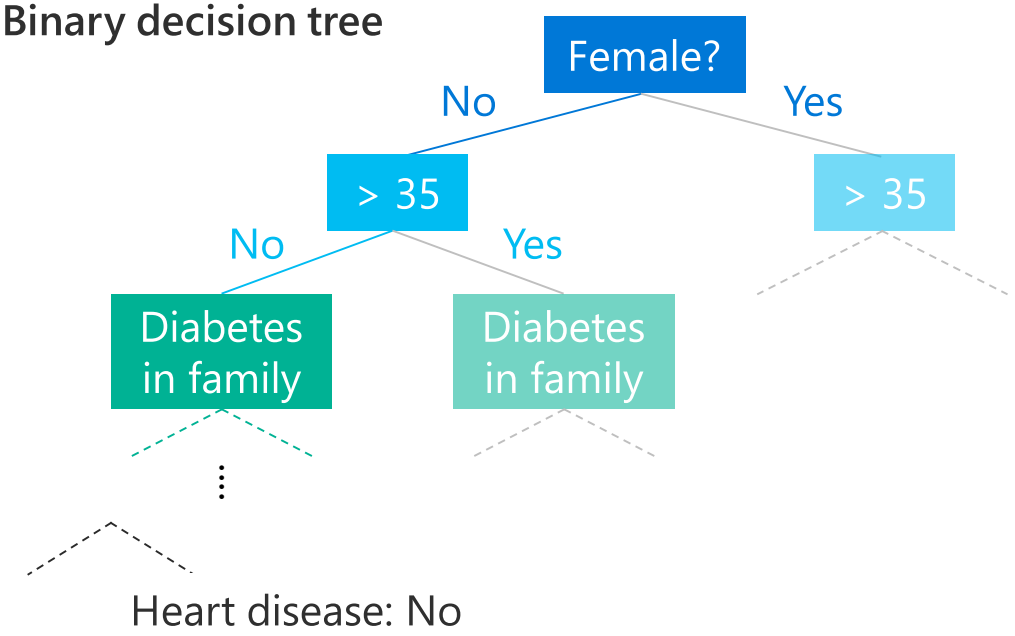
Guarantee: attacker can only observe encrypted communication

Preventing indirect information leaks

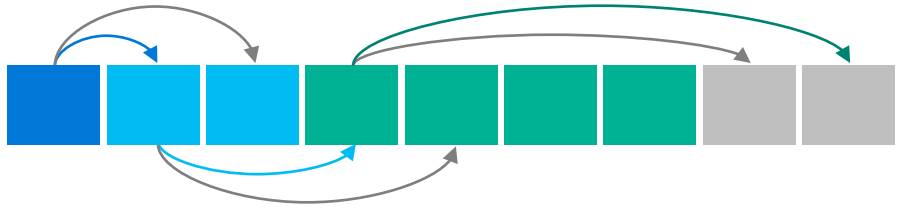
Problem: memory/disk access patterns may leak information

Solution: use compiler and hardened libraries that prevent leaks with data oblivious primitives

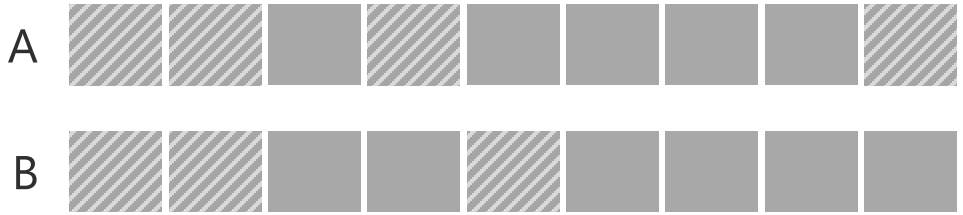
Binary decision tree



Memory



Accesses from 2 predictions



Demo:
Oblivious computing

Example confidential computing scenarios

Always encrypted storage with SQL Server

Enabling scalable and confidential blockchain networks with Coco Framework

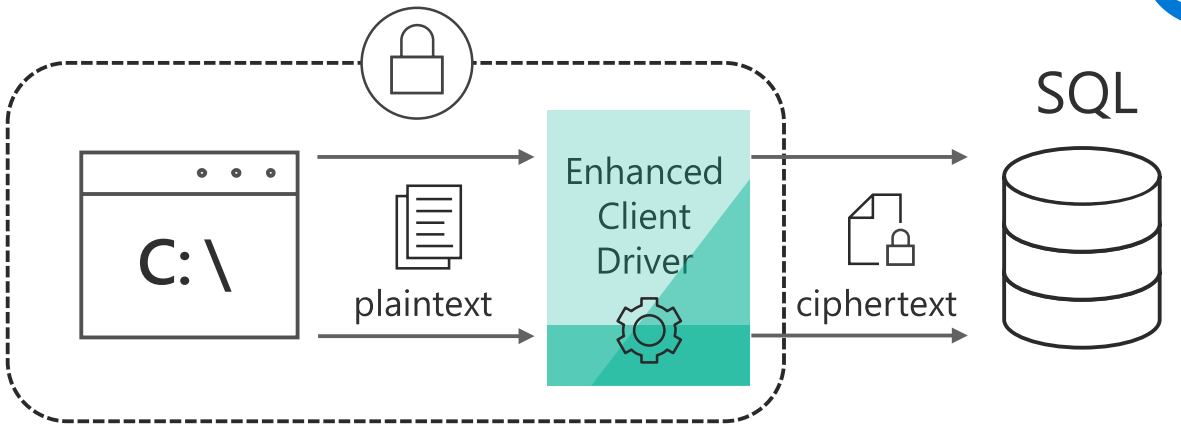
Financial data processing

Secure multi-party machine learning

SQL Always Encrypted

Protects sensitive data **in use** from high-privileged yet unauthorized SQL users both on-premises and in the cloud

Current GA version in SQL Server 2016/17 and Azure SQL DB



Client side Encryption

Client-side encryption of sensitive data using keys that are ***never*** given to the database system

Encryption Transparency

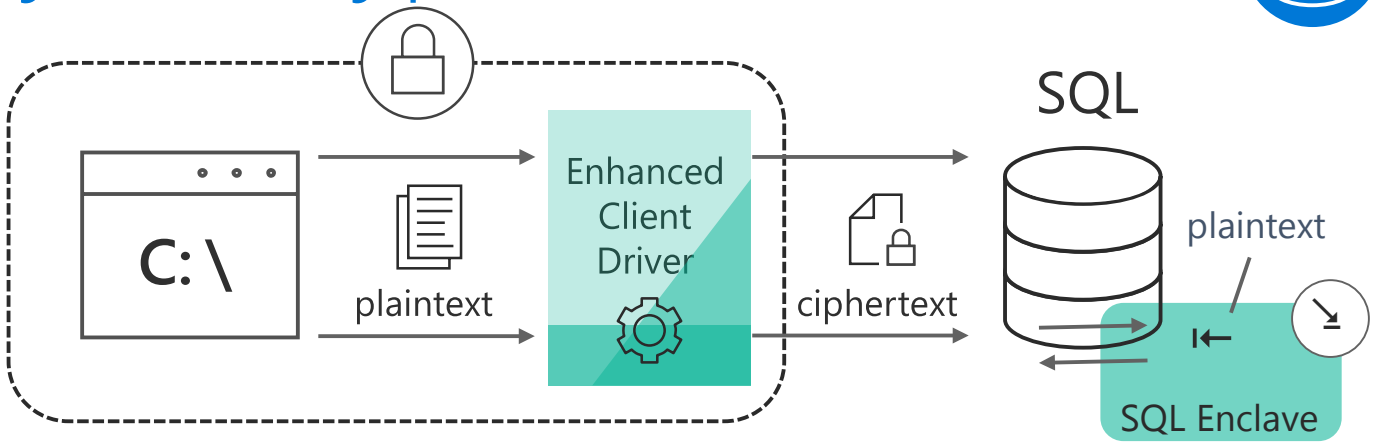
Client driver transparently encrypts query parameters and decrypts encrypted results

Queries on Encrypted Data

Support for equality comparison, including join, group by and distinct operators via deterministic encryption

Confidential SQL Always Encrypted

Protects sensitive data in use **while preserving rich queries and providing in-place encryption**



Secure computations inside SQL Enclave

SQL Server Engine delegates operations on encrypted to the SQL Enclave, where the data can be safely decrypted and processed

Rich Queries

pattern matching (LIKE), range queries (<, >, etc.), sorting, type conversions, support for non-bin2 collation, and more

In-place Encryption

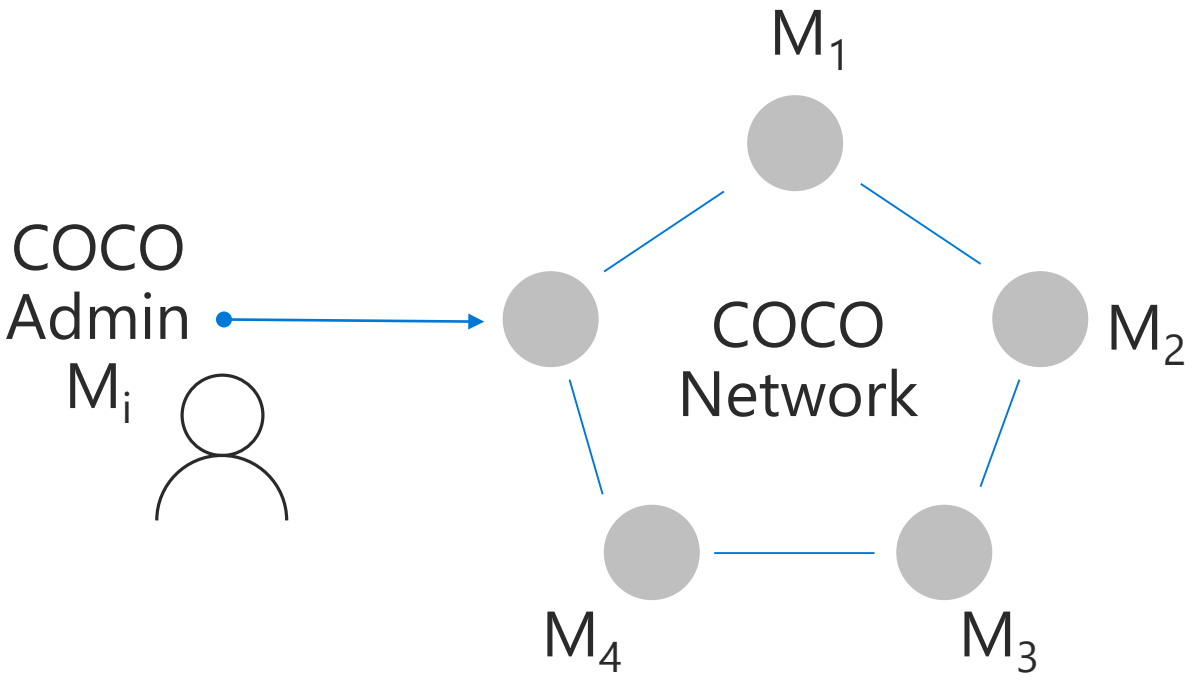
SQL Enclave can perform initial data encryption and key rotation, without moving the data out of the database

Coco Framework: Confidential Consortium Blockchain Framework

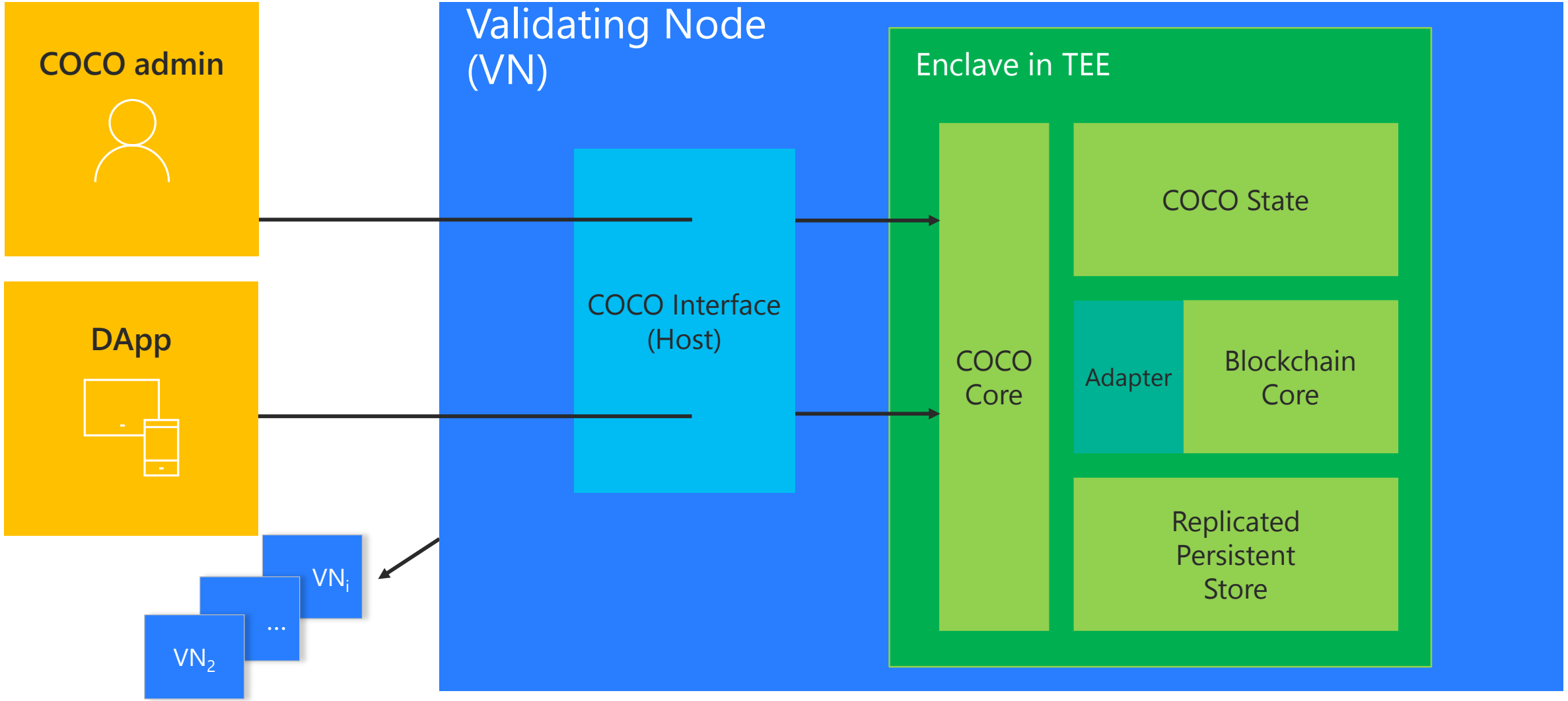
Open-source framework that enables high-throughput (~100x), fine-grained confidentiality, and consortium governance for blockchain

Creates a trusted network of physical nodes on which to run a distributed ledger, providing secure, reliable components for the protocol to use

Through the use of TEEs able to simplify consensus and transaction processing



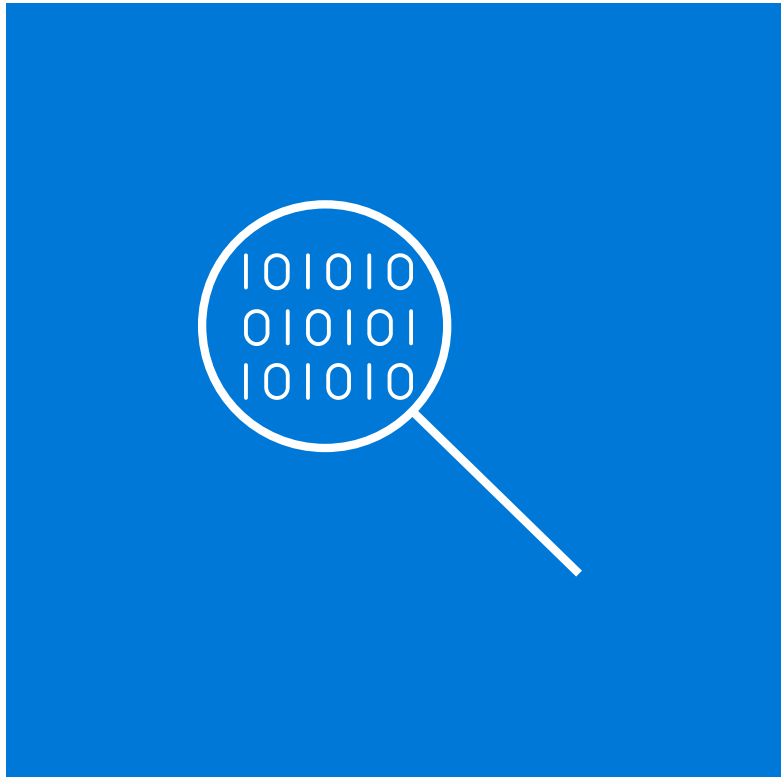
Coco Framework architecture



Coco Framework: Confidentiality model for Ethereum

Coco FX disables access to transaction and block level information

Ethereum smart contract enforces access control rules by verifying address of caller



Smart contract access control

```
98      // Only admins can authorize other users or admins
99
100     function addAuthorizedReader(address a) onlyAdmins
101     {
102         allowedReaders[a] = true;
103         userAuthorized(a);
104     }
105
106     function addAuthorizedWriter(address a) onlyAdmins
107     {
108         allowedWriters[a] = true;
109         userAuthorized(a);
110     }
111
112     function addAdmin(address a) onlyAdmins
113     {
114         admins[a] = true;
115         userAuthorized(a);
116     }
117
118     function removeAuthorizedReader(address a) onlyAdmins
119     {
120         allowedReaders[a] = false;
121         userForbided( a);
122     }
123
124     function removeAuthorizedWriter(address a) onlyAdmins
125     {
126         allowedWriters[a] = false;
127         userForbided(a);
128     }
129
130     function removeAdmin(address a) onlyAdmins
131     {
132         admins[a] = false;
133         userForbided(a);
134     }
135
136 }
```

Smart contract access control (cont)

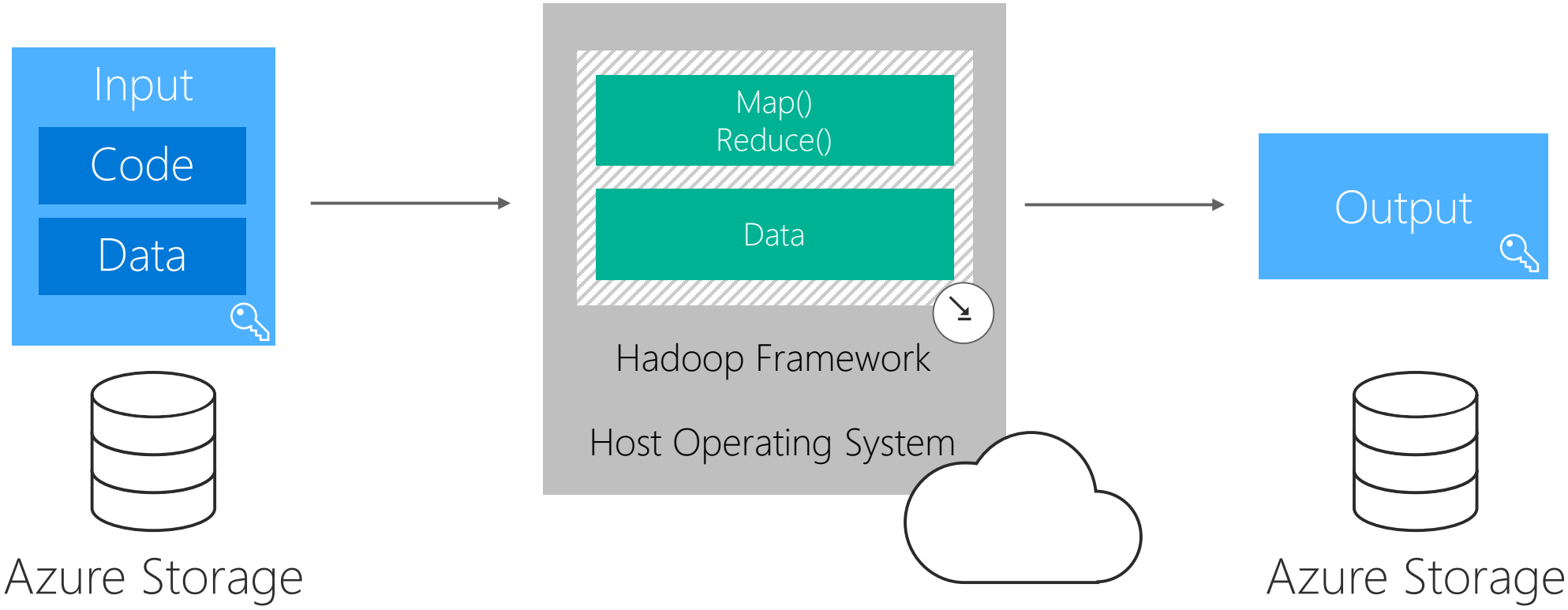
```
pragma solidity ^0.4.0;

/* A sample contract that restricts access to a map of values.  A user is represented by
 * an Ethereum address they own.
*/

contract Vault
{
    // Each access is logged and visible only to authorized users
    event vaultReadAccess(address byWhom);
    event vaultWriteAccess(address byWhom);
    event userAuthorized(address who);
    event userForbided(address who);

   /* The map that is access restricted and the lists of users that are permissioned to
     * read and write.  Note that these members are not public and can only be accessed
     * by code in this contract
     */
    mapping (bytes32 => string) secretsMap;

    mapping (address => bool) allowedReaders;
    mapping (address => bool) allowedWriters;
    mapping (address => bool) allowedEventViewers;

    // Admins can authorize users and add other admins
    mapping (address => bool) admins;

    // Check
    function check(bytes32 hash, uint8 v, bytes32 r, bytes32 s, bytes32 topic) returns (bool){
        address sender = ecrecover(hash, v, r, s);

        bytes32 event1Topic = keccak256("vaultReadAccess(address)");
        bytes32 event2Topic = keccak256("vaultWriteAccess(address)");
        bytes32 event3Topic = keccak256("userAuthorized(address)");
        bytes32 event4Topic = keccak256("userForbided(address)");

        if (topic == event1Topic){
            //`allowedReaders` can access event: `vaultReadAccess`
            return allowedReaders[sender];
        }else if (topic == event2Topic){
            //`allowedWriters` can access event: `vaultWriteAccess`
            return allowedWriters[sender];
        }else if (topic == event3Topic || topic == event4Topic){
            //`admins` can access event: `userAuthorized`, `userForbided`
            return admins[sender];
        }else {
            throw ;
        }
    }
```

```
    // Contract is created with an initial set of authorized users who can read, write and view events
    function Vault(address[] initialAdmins)
    {
        for (uint i = 0; i< initialAdmins.length;i++){
            admins[initialAdmins[i]] = true;
        }
    }


    // A read call made via eth_call is not signed so we require the caller supply an
    // ECDSA signature to authenticate themselves
    modifier onlyAllowedReaders(bytes32 hash, uint8 v, bytes32 r, bytes32 s)
    {
        address reader = ecrecover(hash, v, r, s);
        if (!allowedReaders[reader])
            throw;
        _;
    }

    modifier onlyAllowedWriters
    {
        if (!allowedWriters[msg.sender])
            throw;
        _;
    }

    modifier onlyAdmins
    {
        if (!admins[msg.sender])
            throw;
        _;
    }

    // Modifier onlyAllowedReaders ensures only users authorized to read can call this function
    function read(bytes32 hash, uint8 v, bytes32 r, bytes32 s, bytes24 key) onlyAllowedReaders(hash, v, r, s) constant returns(string)
    {
        vaultReadAccess(msg.sender);  // Generate an event for notification
        return secretsMap[key];
    }

    // Modifier onlyAllowedWriters ensures only users authorized to write can call this function
    function write(bytes32 key, string value) onlyAllowedWriters
    {
        secretsMap[key] = value;
        vaultWriteAccess(msg.sender);  // Generate an event for notification
    }
```

Demo:
Coco Ethereum versus Ethereum

Azure HDInsight

Process massive amounts of data using open source frameworks such as Hadoop, Spark, Hive, Kafka, etc.



Trusted/Untrusted Code Refactoring for Hadoop

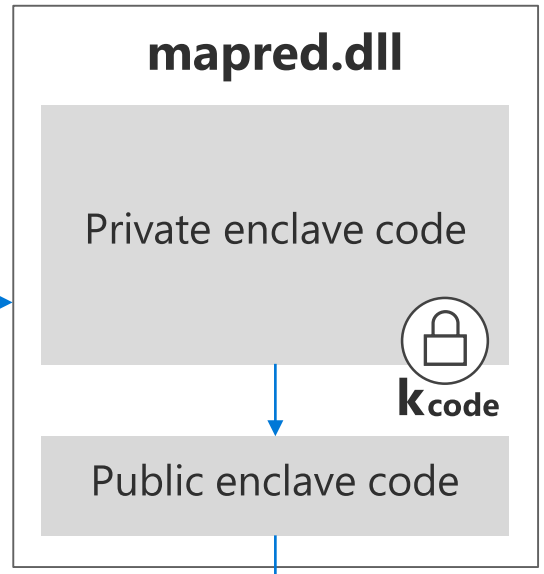
Secret analytics code (C++)

```
void Mapper::map(string k, string v)
{
  /* … */
}

void Reducer::reduce(
 string k, vector<string> v)
{
  /* … */
}
```

+

In-Enclave Lib
(3,300 LLOC)

TCB = mapred.dll
+ SGX processor



Inside enclave/Trusted

Secret user code

Public generic code

- Protocols
- 5,500 LLOC

Outside enclave/Untrusted

Create enclave

Talk to OS

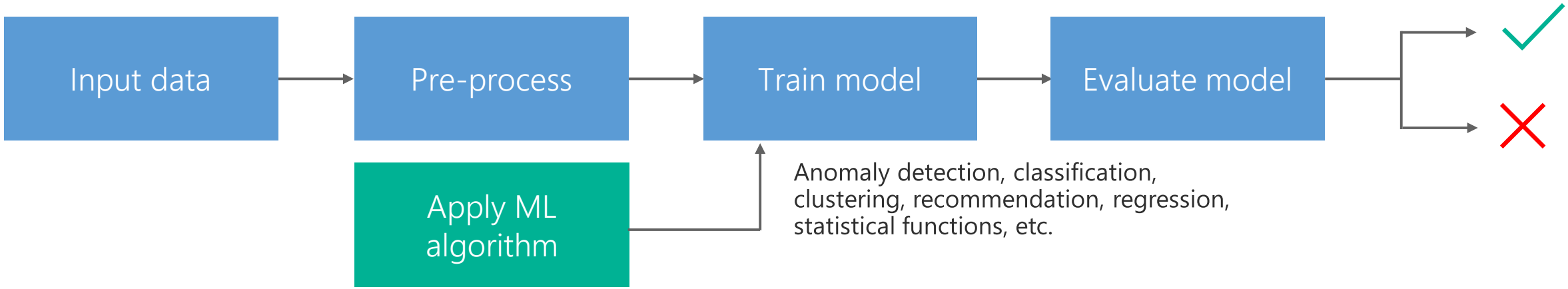
Bind to Hadoop

Azure Machine Learning (ML)

Machine Learning Studio: UI web designer to create an end-to-end ML workflow

Large library of predictive analytic algorithms from which to train models

Modules to support data input, output, pre-processing, and visualizations



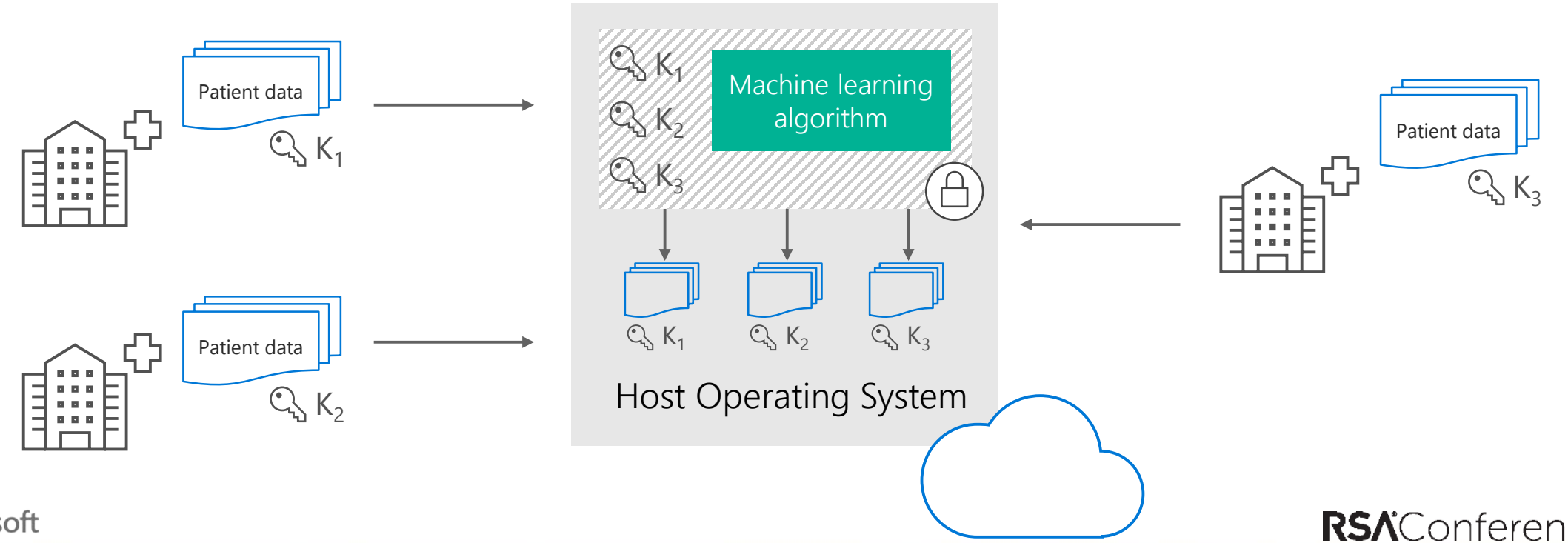
Machine Learning API service: service to deploy prepared ML models at cloud scale and availability

Confidential multi-party machine learning

Partnered health facilities contribute private patient health data sets to train a ML model

Each facility only sees their respective data sets (aka no one, not even cloud provider, can see all data or trained model, if necessary)

All facilities benefit from using trained model



Demo:
Confidential multi-party ML

Summary

Confidential computing in the cloud is in its **early stages**

Microsoft is driving the direction & adoption of newer trusted execution environments in the cloud

Azure is empowering new secure business scenarios in the cloud

References

Blockchain with Coco Fx:
http://aka.ms/cocopaper

Multi-party machine learning:
https://www.microsoft.com/en-us/research/wp-content/uploads/2016/07/paper.pdf

SQL Server with Haven:
https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/osdi2014-haven.pdf

Map/reduce with VC3:
https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/vc3-oakland2015.pdf

Preventing enclave information leaks:
https://people.eecs.berkeley.edu/~rsinha/research/pubs/pldi2016.pdf

Using side-channel page faults to extract JPG images:
https://www.microsoft.com/en-us/research/wp-content/uploads/2017/06/atc17-final230.pdf