VMware Certifier Framework for Confidential Computing

Anchoring End-to-end Trust and Security and Beyond

John Manferdelli

jmanferdelli@vmware.com

The Data Protection Problem

For the Data Economy and other “trustworthy” services

I want to run programs and guarantee the confidentiality and integrity of the programs and the data they use, no matter where they run, at the edge, in a “private cloud”, in a “public” cloud” or at a “partner” site.

Programs and data must be protected from

1. Accidental or malicious misconfiguration
2. Insider abuse including modifying the program.
3. Any unauthorized disclosure or modification of the data
4. Disclosure of critical secrets and keys affecting security (including the data)
5. Violation of data use rules by multiple owners’ whose data is present in the data set

Standard Applications

- Hardware secure module
- Secure Key Store and token generation
- Secure Motion planning as a service
- Secure collaborative machine learning
- Secure Auctions
- Secure Kubernetes container management (via secure Spiffie/Spire)
- Secure federated identity management
- Secure Database
- gRpc
- Secure Document sharing
- Secure sensor collection
- Caching services
- Standard platform components (storage, time, IAM)
  - Hashicorp
  - Azure
  - Amazon services

Confidential Computing Provides the Foundation

Four capabilities of a Confidential Computing:

- **Isolation.** Program address space and computation.
- **Measurement.** Use cryptographic hash to create an unforgeable program identity.
- **Secrets.** Isolated storage and exclusive program access. (aka, “sealed storage”).
- **Attestation.** Enable remote verification of program integrity and secure communication with other such programs.

Isolation and measurement
- Program address space isolated
- Program hashed to give non-forgeable identity

Secrets
- Seal: protect a secret for this measurement
- Unseal: restore a secret for this measurement

Attestation
- Statement signed by a trusted party (HW) that specifies
  - program identity (measurement) program
  - hardware protection (isolation, integrity, confidentiality) guarantees
  - Statement attributable to isolated entity

vmware®

How does Confidential Computing achieve security

Program Isolation and integrity: Hardware reads programs into memory and isolates them from other programs including the VMM

Identifying programs that are trusted: Hardware takes a cryptographic hash of the program before it starts and remembers it. This is called the program measurement.

Protecting basic keys: Isolated program generates authentication keys and storage keys. No one sees them but the program. Program seals these secrets so they can be recovered (by Unseal) at restart

Enforcing policy: A public key is embedded in the program. Program will (and can only) perform actions the corresponding private key authorized by signing instructions (policy)

Unforgeably authenticating programs: Program takes its public authentication key and asks hardware to attest to it. This means the hardware signs a statement naming the public key and the program measurement.

How does Confidential Computing achieve security

Trusting a program's public key: Using the attestation and other signed evidence, another program (the "relying program") can verify the original program (the "requesting program") is trustworthy (by its measurement) and now has a foolproof way to authenticate it (by it's public key). This is the "trust decision." This results in an X509 Certificate ("Admissions Certificate").

Securely sending data and rules to authenticated programs: The relying program and requesting program can use their now authenticated public keys to open authenticated, encrypted, integrity protected channels.

Protecting transmitted data: Data is only sent to trusted programs over secure channels.

Protecting stored data: If a program wishes to store data, it makes up a protection keys, seals those keys (and stores the sealed results), encrypts and integrity protects the data with those keys and stores the encrypted files.

That's all we need

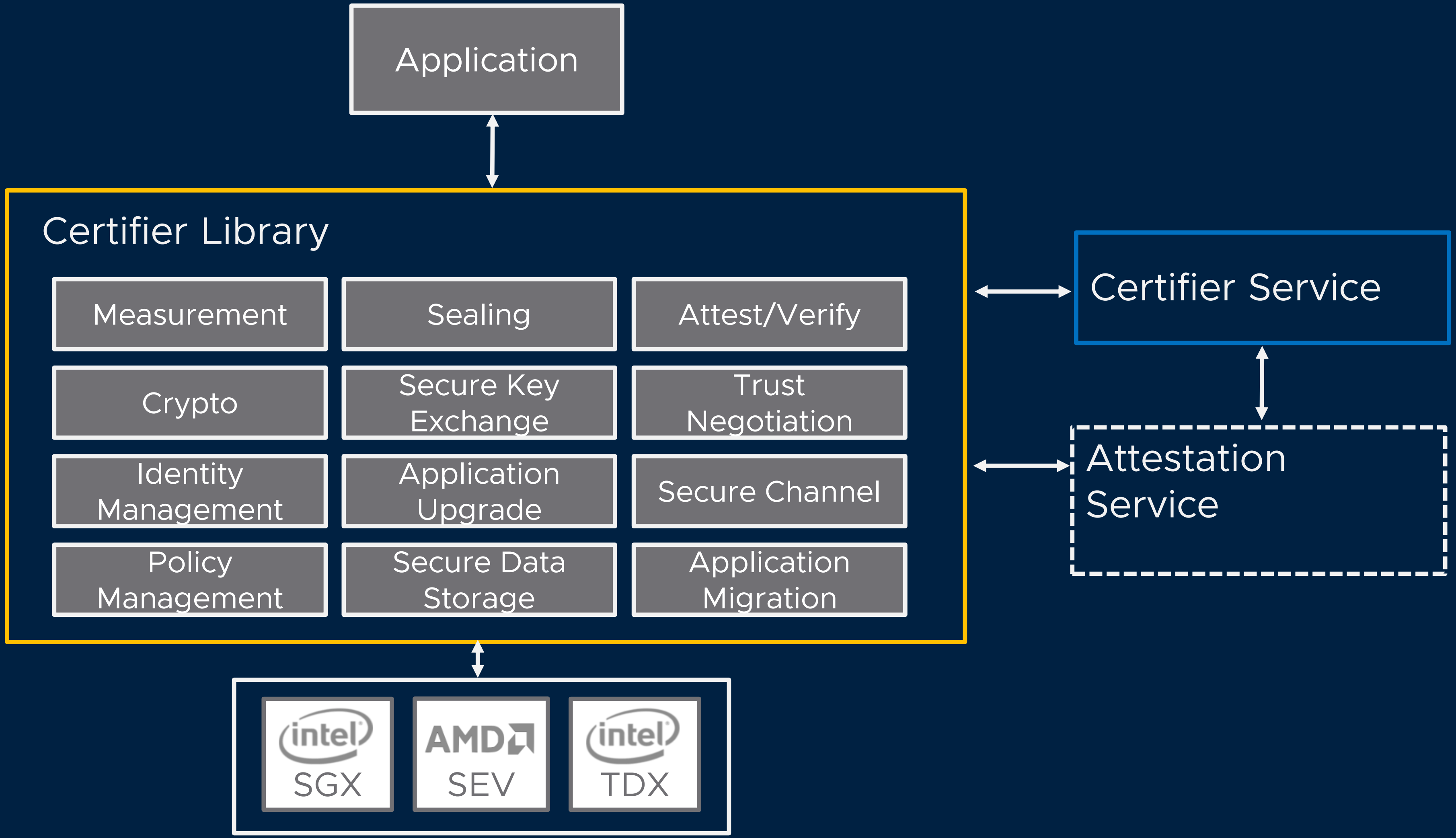
What’s hard about this?

- Lots of keys to keep track of
- Policy can be complicated and general, it must be universally understood by all “trusted programs”
- Policy must be verified by a trusted program with absolute assurance
- All the program secrets and data must be securely stored (no cheating)
- Secure channel must be established unforgeably identifying both ends
- Must work on all Confidential Computing platforms
- Must not require extensive changes to an already (well-written) application
- Must not interfere with scalable deployment models (including adding new programs)
- Must be scalable
- Security code is hard to get right

Made easy using The Certifier Framework for Confidential Computing

The Certifier Framework for Confidential Computing

Taking the devil out of the details in using Confidential Computing



vmware

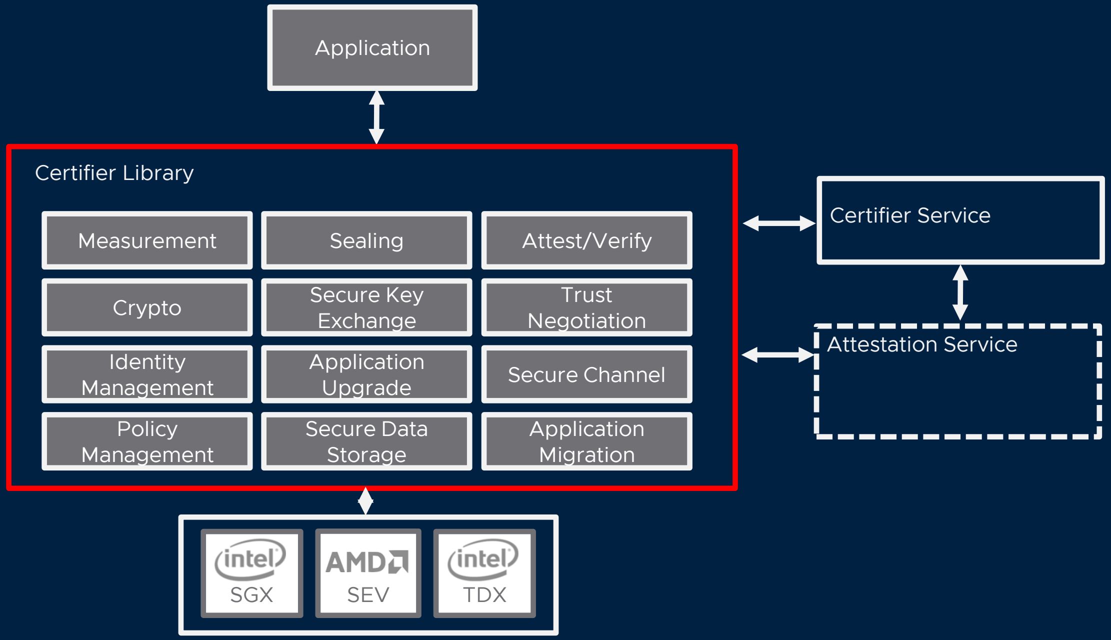
Certifier Library

Goals

- Turn developer task to use CC from developing thousands of lines of code to a dozen calls.
- Associate policy with application rather than embedding it
- Enable management policy bilateral and service based scalable application management that supports secure sharing

Unified interface to confidential computing primitives

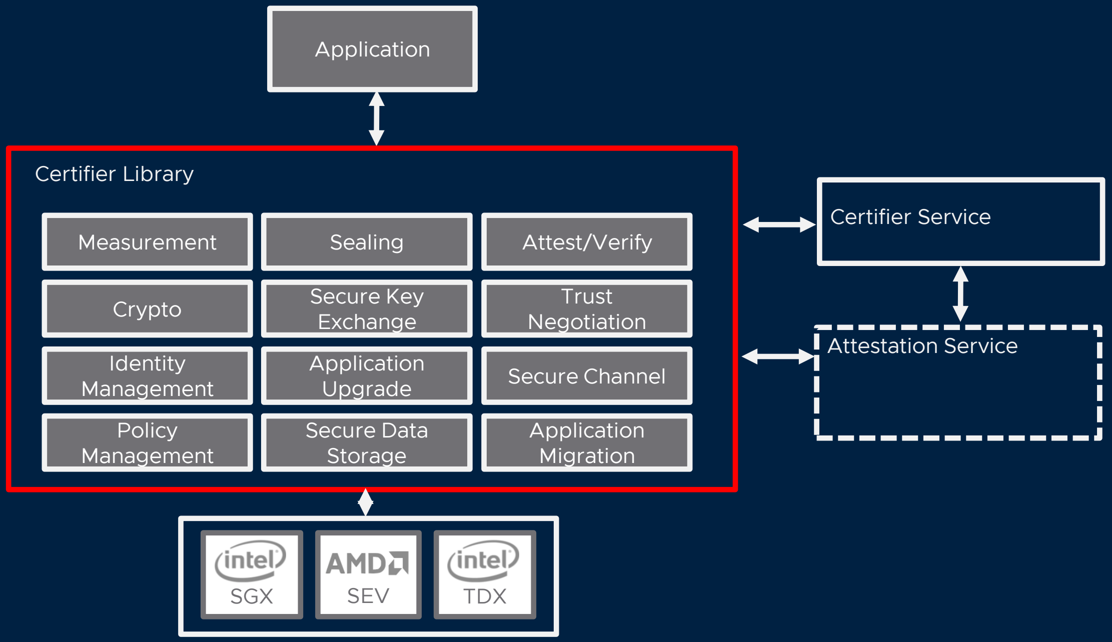
- Measure, Seal, Unseal, Attest, Verify attestation



Certifier Library

Provides simple primitives to

1. Implement secure, authenticated storage and recovery of critical confidential container data
2. Securely exchange keys with another trusted confidential container
3. Upgrade programs
4. Determine whether another program should be trusted: Implements trust negotiation using simple uniform policy management (Certification).
5. Establish secure channel between mutually trusted containers using X509 ("admission") cert.
6. Other helper functions



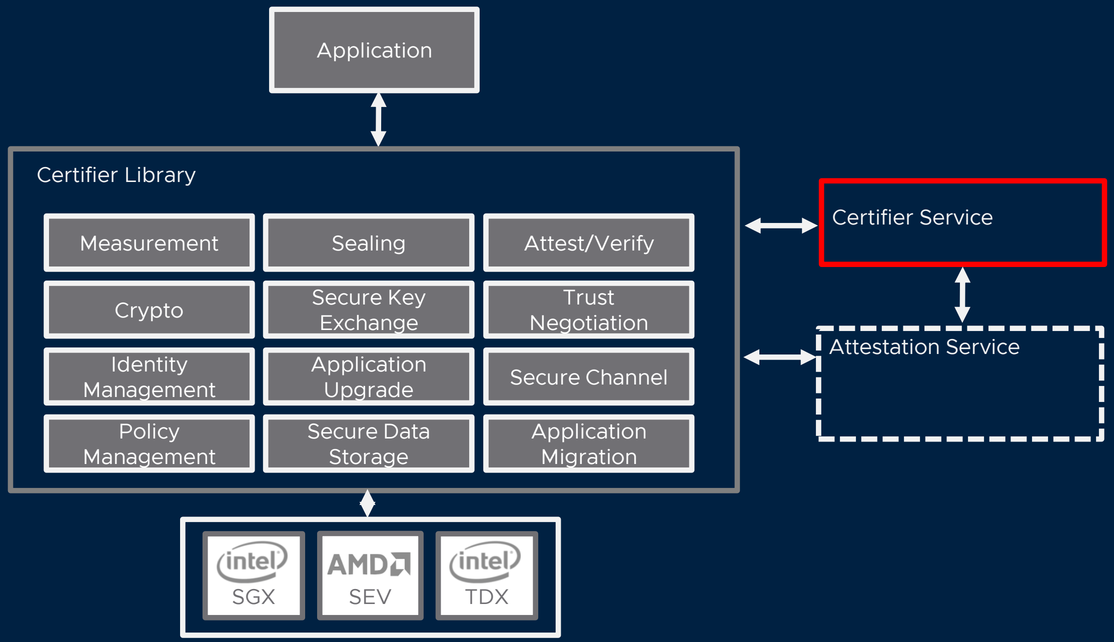
But wait there's more: Same interface for all platform enforcers (SGX, SEV, TDX, …)

vmware

Certifier Service

“Centralized” management for a security domain

- Policy Driven
- Admit new components without changing old ones
- Application upgrade without changing other applications
- Facilitates data migration and sharing in a domain
- Enforce security domain wide policy
  – Machine capabilities



SCALABLE DEPLOYMENT SUPPORTING ALL ENVIRONMENTS

Policy driven trust management

Policy

- Rooted in policy key which is in application and part of its measurement
- Policy expressed in signed claims
- Claims are human readable
- Policy language is general
- Supports delegation
- Other policy languages can be used (e.g.-OPA)

Example

```
1. Key[rsa, policyKey,
a5fc2b7e629fbbfb04b056a993a473af3540bbfe] is-
trusted

2. Key[rsa, policyKey, ... ] says
Measurement[a051a41593ced366462caea3928306287429
43c3e81892ac17b70dab6fff0e10] is-trusted

3. Key[rsa, policyKey, ...] says Key[rsa,
platformKey, ...] is-trusted-for-attestation

4. Key[rsa, platformKey, ...] says Key[rsa,
attestKey, ...] is-trusted-for-attestation
```

Policy properties

Example

```
1. Key[rsa, policyKey, …] is-trusted

2. Key[rsa, policyKey, ... ] says
Measurement[…]  is-trusted

3. Key[rsa, policyKey, ...] says
Key[rsa, platformKey, ...] is-
trusted-for-attestation

4. Key[rsa, platformKey, ...] says
Key[rsa, attestKey, ...] is-trusted-
for-attestation
```

Native Policy Language (SPKI/SDSI)

Evidence has same format (although others, like certificates, can be used), including attestation:

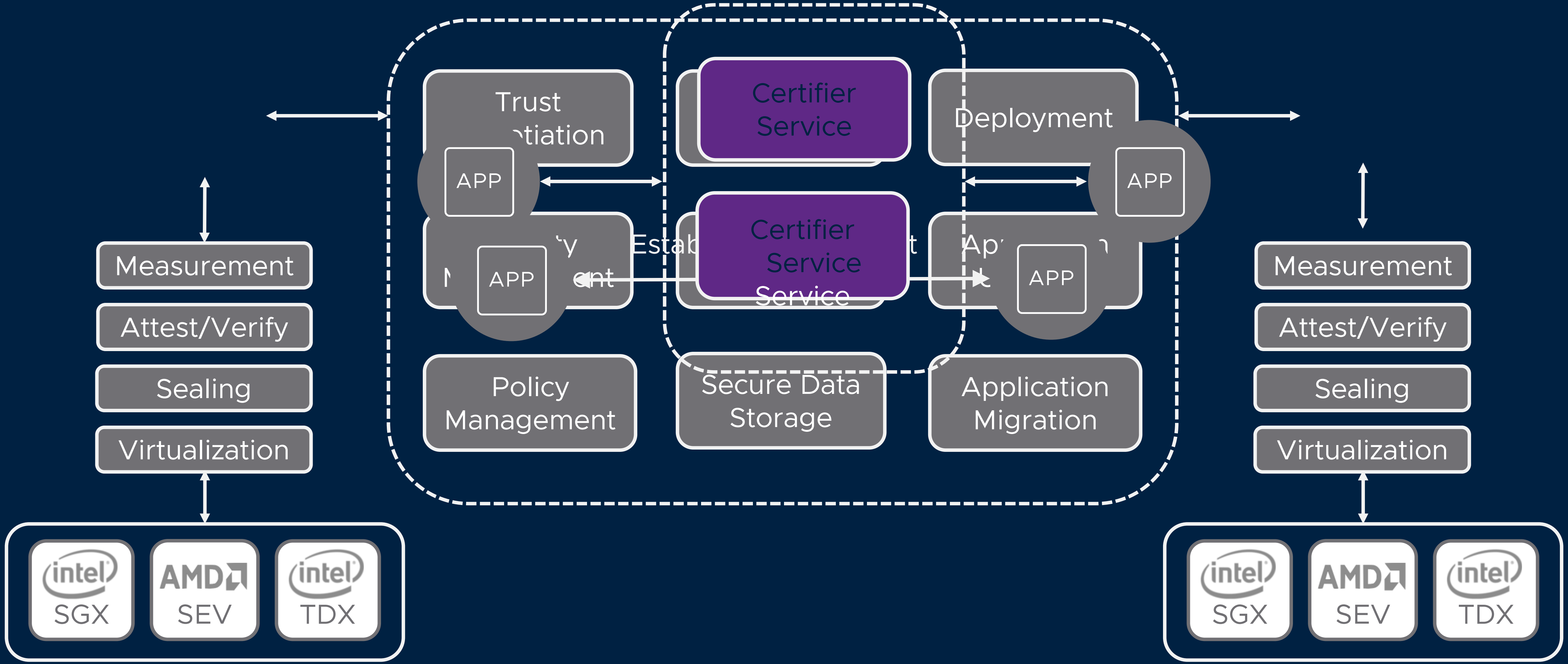
```
Key[rsa, attestKey, ...] says Key[rsa,
program-auth-key, ...] speaks-for
Measurement[...]
```

Proof

- Proof is also human readable!
- Same format
- Authentication-key is-trusted-for-authentication
- Also encoded in X509 certificate naming measurement and key

vmware®

Certifier takes devil out of the details



Proofs from the Certifier Service

1. Key[rsa, policyKey, ...] is-trusted and Key[rsa, policyKey, ...] says Measurement[a051a41593ced366462caea392830628742943c3e81892ac17b70dab6fff0e10] is-trusted, imply via rule 3, Measurement[...] is-trusted

2. Key[rsa, policyKey, ...] is-trusted and Key[rsa, policyKey, ...] says Key[rsa, platformKey, ...] is-trusted-for-attestation, imply via rule 5, Key[rsa, platformKey, ...] is-trusted-for-attestation

3. Key[rsa, platformKey, ...] is-trusted-for-attestation and Key[rsa, platformKey, ...] says Key[rsa, attestKey, ...] is-trusted-for-attestation, imply via rule 5, Key[rsa, attestKey, ...] is-trusted-for-attestation

4. Key[rsa, attestKey, ...] is-trusted-for-attestation and Key[rsa, attestKey, ...] says Key[rsa, program-auth-key, ...] speaks-for Measurement[...], imply via rule 6, Key[rsa, program-auth-key, ...] speaks-for Measurement[...]

5. Measurement[...]is-trusted and Key[rsa, program-auth-key, ...] speaks-for Measurement[...], imply via rule 1, Key[rsa, program-auth-key, ...] is-trusted-for-authentication

Proved: Key[rsa, program-auth-key, ...] is-trusted-for-authentication

5 steps

Extended Example

Organizations want to analyze shared data about fraud on them.

They want to spot fraud trends based on a large data set from many similar organizations

They do not want their data disclosed

They do not want other participating organizations to see their data

They don't want regulators to see it either

Each organization must trust that the analysis is fair and honest

No data but aggregated fraud trend must be disclosed

It must run at any organization site or in any cloud

Initial Setup: Banks Collaborate

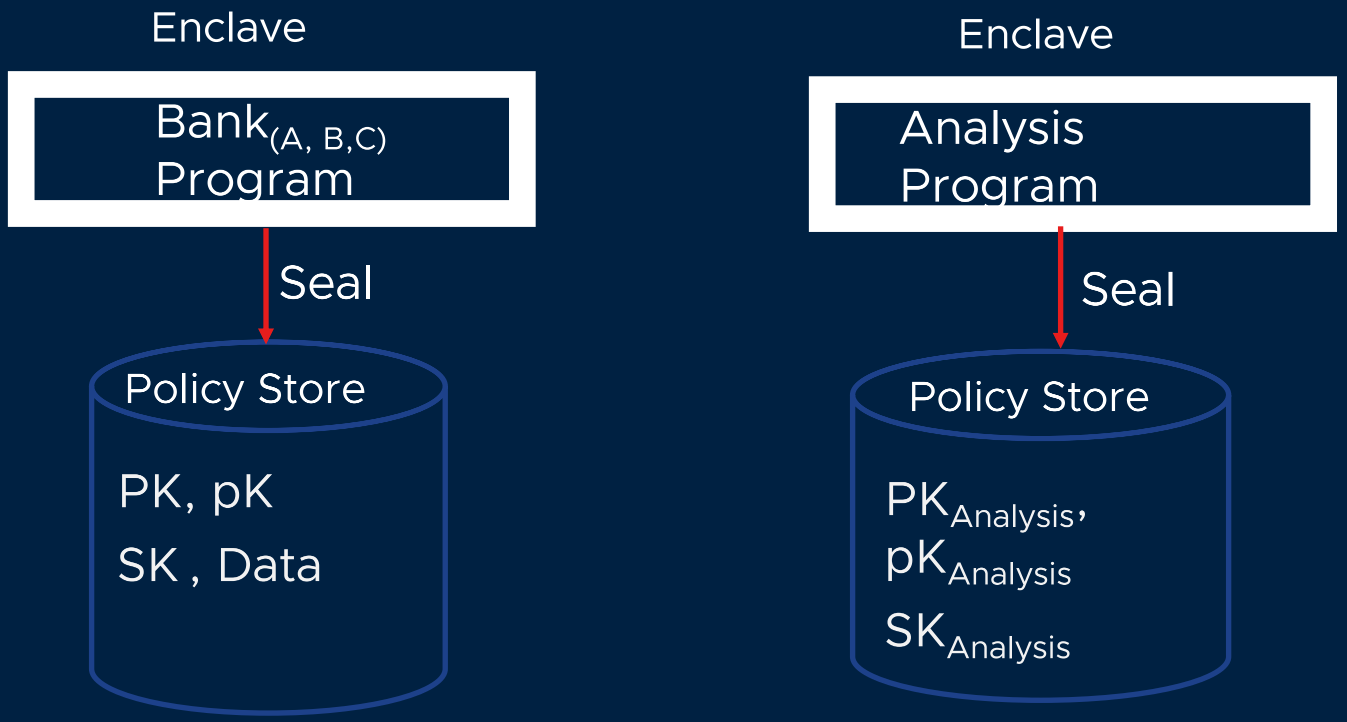
All three parties (A, B, C) get together and agree on the analysis and what can be released from the analysis.

- **Bank (A, B, C) transmission program:** A Confidential Computing Program at each Bank that has the fraud data for that bank
- **Analysis Program:** The analysis program is created/selected by the three parties in collaboration. Each can examine the program to ensure security properties are met.
- **Measurement.** These four programs are measured (M[Analysis], M[Bank A], M[Bank B], M[Bank[C])
- Certifier Service:
  1. Service generates the policy key and deploy public key for inclusion in application
  2. Service acquires root certificates from vendors
  3. Banks use utilities to generate policy (including trusted measurements)
- Banks deploy Certifier Server instances

Analysis and Bank Program Initialization

Initialization:

1. Generate a public/private key pair (PK/pK)
2. Generate (symmetric) storage keys (SK)
3. Save key in policy store and save store using seal for later use
4. Each Bank program fetches data and encrypts it to its enclave

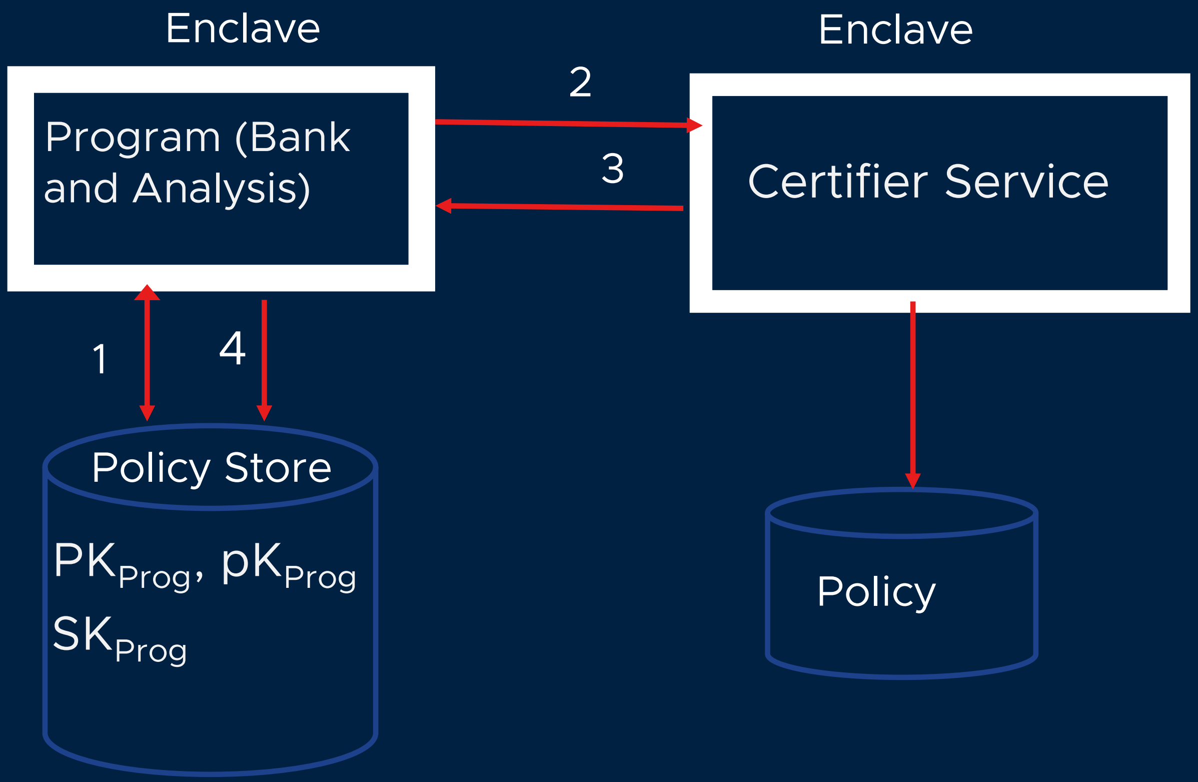


Security Implication

- These Programs and *only* these Programs will have access to their keys and *only* when isolated on a "trusted platform."

vmware

Each Program Gets Certified

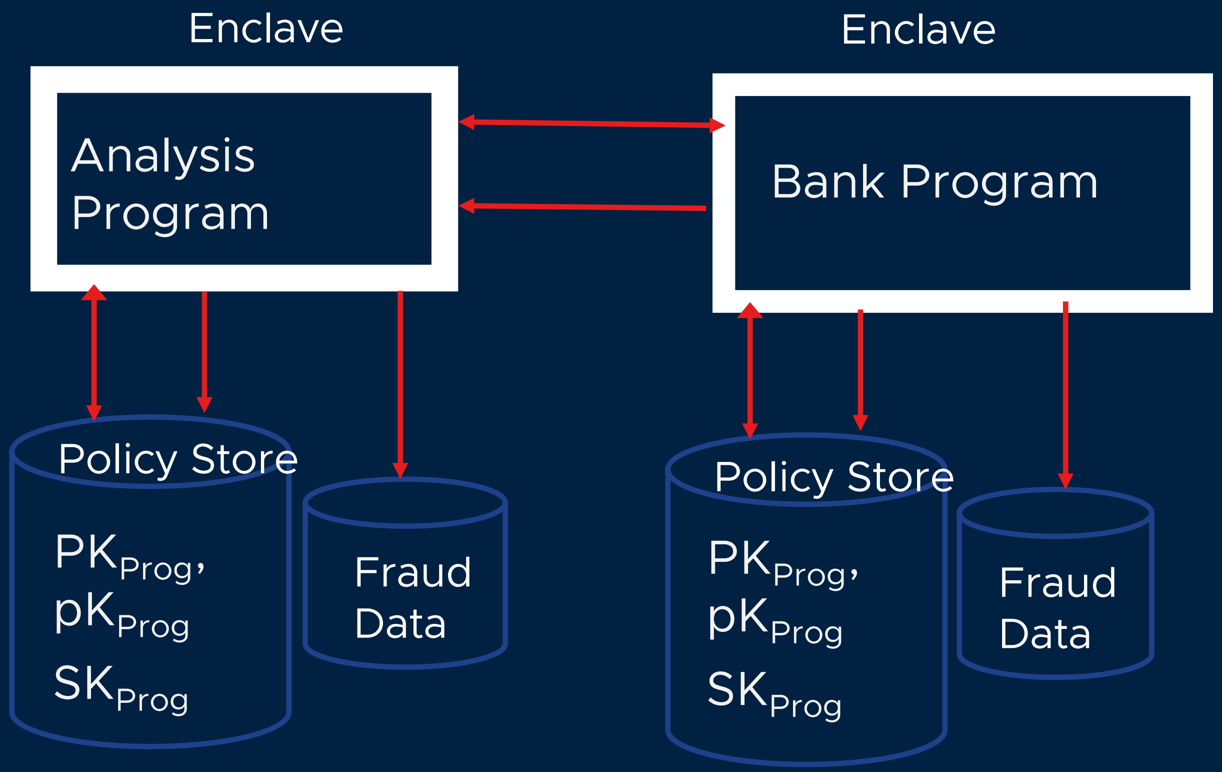


1. Program unseals its encrypted keys
2. Program sends attestation with its program-auth-key and evidence to Certifier Service.
3. Certifier Service verifies evidence against policy and, if compliant, returns X509 certificate (the “Admissions Certificate”) naming the program measurement and program-auth-key .
4. Program stores it Admission certificate in the policy store and saves the store

Security implication:

1. Each Program has an Admissions Certificate
2. Only the program can recover its private key
3. Program data can be recovered only by the program

Data Provisioning

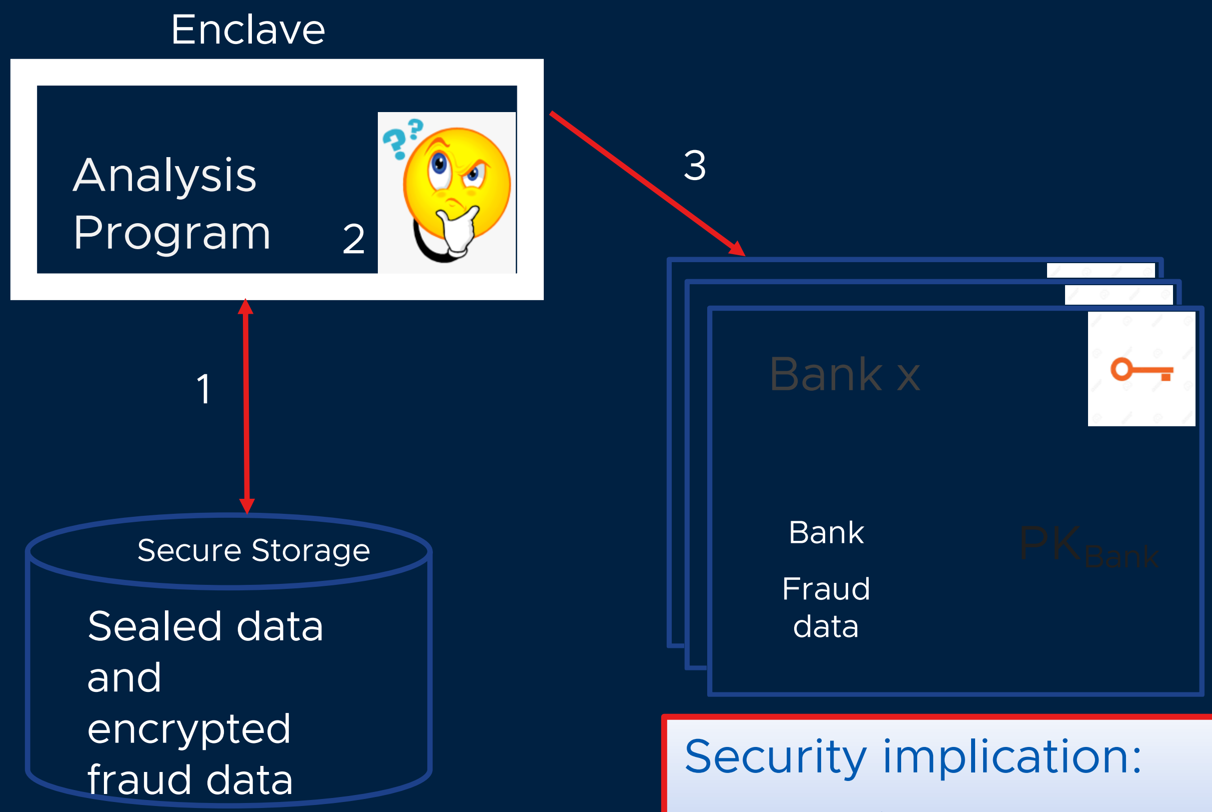


1. Programs unseals their encrypted keys
2. Programs exchange Admissions certificates and establish secure channel
3. Bank program sends fraud data to Analysis program
4. Analysis program stores aggregated fraud data

Security implication:

1. Each Program has an Admissions Certificate
2. Only the program can recover its private key
3. Program data can be recovered only by the program

Analysis Program Performs Fraud Analysis and Reports to Banks



1. Analysis Program unseals its protected data (w/ $PK_{Analysis}$, $pK_{Analysis}$ and $SK_{Analysis}$), and retrieves the encrypted fraud data.
2. Analysis Program decrypts the fraud data and then performs the analysis.
3. Analysis Program forwards the results to each bank (perhaps protecting it with ($PK_{Bank\ A}$, $PK_{Bank\ B}$, $PK_{Bank\ C}$).

Security implication:

1. Data is encrypted at rest and in transit at all times
2. Only Analysis Program can see raw unencrypted fraud data and only in isolated Program environment
3. Analysis Program ensures it only releases approved, aggregated analysis to authorized parties.

vmware®

Certifier Framework for Confidential Computing Status

- Open Sourced!
  - Apache 2.0 licensed
    - Includes sample apps, how to, “Hello world”
  - Beta looking for partners and feedback (This means you!)
  - Plan to submit to standards groups
  - Contains Certifier API Library (4000 LoC, C++), Certifier Service (3000 LoC, Go), utilities, sample applications, tests, documentation, step-by-step instructions

- Supported platforms
  - SGX via open-enclaves
  - SEV-SNP
  - TDX [future]
  - Realms [Future]
  - RISC-V [Future]

Development and Deployment Flow

Development tasks
(Each step corresponds to a Certifier API call)

First time program runs

1. Initialize authentication key and store
2. Obtain attestation naming authentication key
3. Provide attestation and other evidence to certifier service
4. Get admission certificate and add to secure store
5. Securely save store (uses seal)
6. Perform application logic

Thereafter

1. Retrieve secure store (uses unseal)
2. Obtain authentication keys, admission certificate etc from store

Deployment Tasks

1. Embed your policy key
2. Deploy you application as you do now
3. Run application as you do now

Trust policy evaluation flow (certification)

Service Preparation and Deployment

1. Generate your policy key
2. Use tools provided to author security domain policy
3. Add policy on trusted measurements and hardware to service
4. Run Service (multiple instances for resilience)

Service Processing Flow

1. The Certifier Service receives certification request with attestation naming authentication key
2. Verifies policy compliance
3. Logs relevant data
4. Issues Admission Certificate for Security Domain naming measurement and authentication key

vmware®

Summary

- Trustworthy multi-cloud security needs trusted mechanisms to guarantee confidentiality, integrity, and policy enforcement everywhere
  - Confidential Computing provides the mechanisms
  - Without more support, secure Confidential Computing enabled programs can be hard to write, difficult to understand, require extensive application changes and run on only a single targeted hardware platform and can be cumbersome to manage and deploy
  - The Certifier Framework for Confidential Computing fixes that
  - Existing application → scalable, managed Confidential Computing program by adding a dozen calls
  - No deployment changes, works on any provider
  - Simple management and policy enforcement via the Certifier Service
  - It's open source

vmware®

Evolution of Confidential Computing

- Verifiably secure operational properties, including confidentiality, integrity and policy compliance, no matter where program runs
- Critical to multi-cloud security (or any distributed application running anywhere)

Unsafe distributed applications without CC

1. Write the the application correctly
2. Deploy the program safely (no changes)
3. Configure the operating environment correctly
4. Ensure other programs can't interfere with safe program execution
5. Generate and deploy keys safely
6. Protect keys during use and storage
7. Ensure data is not visible to adversaries and can't be changed in transmission or storage
8. Ensure trust infrastructure is reliable
9. Audit to verify this all happened

- App writer/deployer entirely reliant on provider for all security --- unverifiable

Safe distributed applications with CC

1. Write the application correctly
   - For every backend
   - Manage migration
   - Support each providers deployment model
   - Implement all the crypto
   - Implement secure communications and storage
   - Make it scalable and upgradable
2. Implement the trust policy
   - Maintain trust policy
   - Different for every app/deployer
   - Make it scalable

- You can do it but it's technology dependent and a lot of work

Safe distributed applications with Certifier

1. Write the application correctly using VMware APIs
2. Write the trust policy
3. Let VMware deploy for any backend and and manage it!

- Write the application once.
- Only a few dozen lines of code securely handles CC functions
- Trust policy is independent of application.
- Move to another "backend" effortlessly

Attestation Nomenclature

Platform attestation

- Verify hardware and hardware dependency characteristics (hardware model and generation, bios, microcode)
- Example: Classic Intel service attestation for SGX
- Does not interpret ("User Data") or evaluate comprehensive program policy which is left for application developer

Program or full policy attestation

Evaluates full policy compliance including platform characteristics, program-based trust and assures program carries out security domain policy

Certifier Service provides full policy attestation and trust management across all platforms (it can access platform attestation services if desirable)

Beware

- "Attestation Services" typically provide platform attestation for specific hardware (e.g. – Intel SGX platforms)
- Certifier Service provides comprehensive attestation support including platform and full policy attestation
- "Universal" platform attestation is helpful but generally not mandatory (platform characteristics can be verified directly by certifier service.
- Universal policy attestation raises scalability, freshness, security and privacy concerns