CONFIDENTIAL COMPUTING SUMMIT 2023

vmware

Overcoming Barriers to Confidential Computing as a Universal Platform

John Manferdelli, VMware

jmanferdelli@vmware.com

Organized by Opaque and Confidential Computing Consortium

CC Provides a Foundation for Security

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.

CC provides principled security wherever your programs run and wherever your data resides *even if you don’t operate the computers the programs run on.*

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

The Promise of Confidential Computing

Security enablement anywhere (cloud or not) →

- Standard platform components (key store, storage, time, IAM)
- Secure shared data access (Regulators: GDPR, Health, Finance)
- Safe program execution (“A safe place to stand in the cloud”)
- Zero Trust

Secure privacy preserving service enablement (Data Economy) →

- Secure collaborative machine learning
- Secure Motion planning as a service
- Secure Auctions

Secure infrastructure management →

- Secure Kubernetes container management
- Secure Document sharing

Platforms for sensitive edge services →

- Edge sensor collection
- Caching services and the “extended internet”

Barriers to Confidential Computing Adoption

“In the future, all programs will be Confidential Computing Programs” -- Intel

Hardware

- TEE availability
- Programs are not “portable” across platform technologies

Software

Writing or converting programs to CC is difficult!

- Different code for different programs
- Complicated support code raises a security problem and you can’t start right away
- No “copy and paste” to get started with a secure program
- Cooperating programs require flexible management infrastructure to support many program providers and different security requirements.
  - Trust policy often embedded in program (Bad!)
  - Policy difficult to write, understand or audit

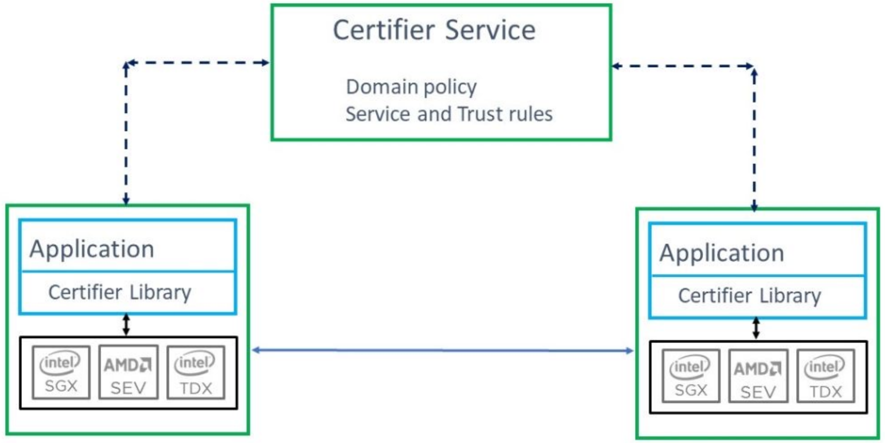
What is the “Linux” for Confidential Computing?”

Software Tedium: All that Work

- Generating, rotating and managing lots of keys
- Authoring, managing and enforcing program policy universally understood by all “trusted programs”
- Binding security policy to pre-authored program
- Verifying policy compliance with absolute assurance
- Securely storing and recovering secrets and data
- Securely communicating with other unforgeably identified Confidential Computing Programs
- Operating on different Confidential Computing platforms without application changes
- Rapid CC enablement of existing “well written” programs
- Preserving existing deployment models
- Providing scalable support managing related distributed components (including upgrade and new components)
- Enabling features with secure code and appropriate, agile logging, encryption and authentication primitives

- Earlier SDK’s help but don’t do it all.

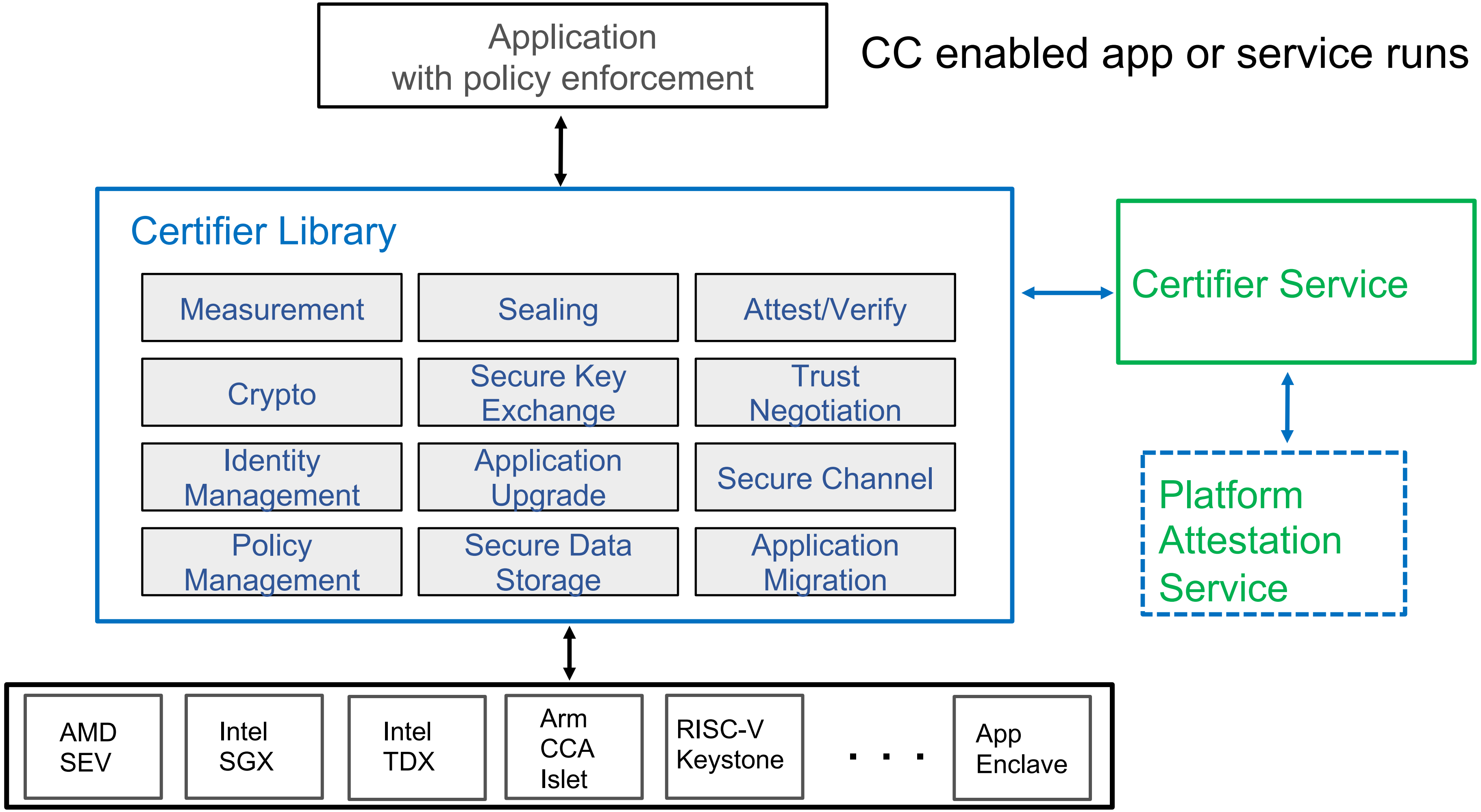
The Certifier Framework



- Open-source framework for Confidential Computing
- Decreases time to build applications
- Avoids need to port application
  - Already supports SGX (via Open Enclaves or Gramine), AMD-SEV-SNP, Arm CCA, RISC-V Keystone, and soon TDX.
- Simple API for most applications but complete support for “edge” cases
- Simple and secure management: simple “embarrassingly parallel” server support infrastructure to support policy management
  - No deployment changes
  - Audited comprehensible policy
  - Efficient introduction of new components and upgrades
- Enable new services and use models for Data Economy, high security applications and regulatory regimes

Certifier Framework Architecture

Taking the devil out of the details



Application
with policy enforcement

CC enabled app or service runs everywhere

Certifier Library

Measurement	Sealing	Attest/Verify
Crypto	Secure Key Exchange	Trust Negotiation
Identity Management	Application Upgrade	Secure Channel
Policy Management	Secure Data Storage	Application Migration

Certifier Service

Platform Attestation Service

AMD SEV · Intel SGX · Intel TDX · Arm CCA Islet · RISC-V Keystone · . . . · App Enclave

Service runs anywhere

Under your desk

In the cloud

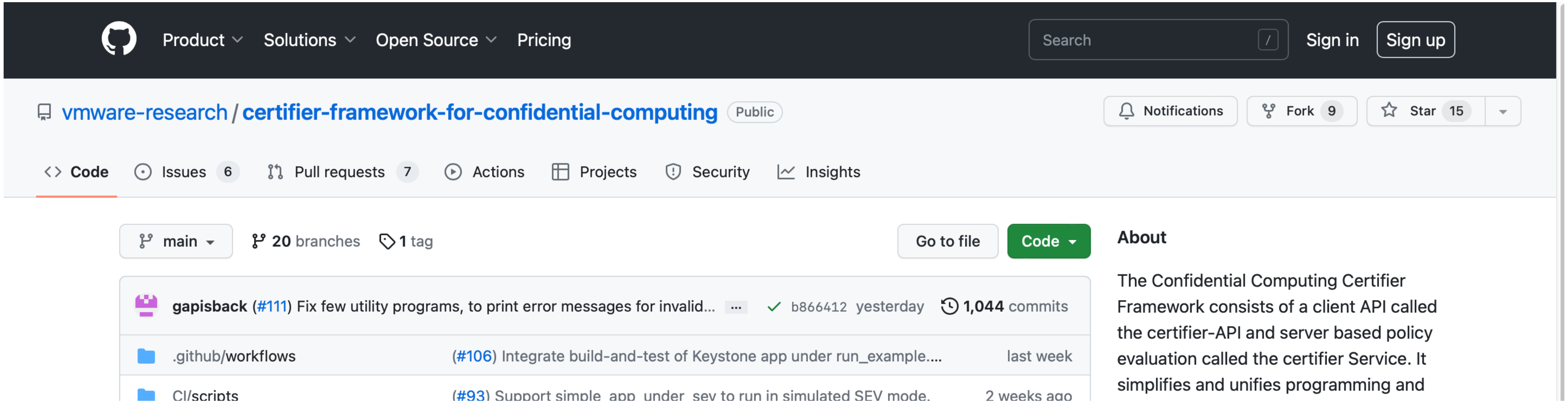
At a "service provider"

Selectable security control

Join the party

Open-Source project

- github.com/vmware-research/certifier-framework-for-confidential-computing
- Apache license
- Contributions and new contributors are welcome
- Make Confidential Computing an open Universal Platform



vmware-research / certifier-framework-for-confidential-computing Public

gapisback (#111) Fix few utility programs, to print error messages for invalid... ✓ b866412 yesterday 1,044 commits

.github/workflows (#106) Integrate build-and-test of Keystone app under run_example.... last week

CI/scripts (#93) Support simple_app_under_sev to run in simulated SEV mode. 2 weeks ago

About

The Confidential Computing Certifier Framework consists of a client API called the certifier-API and server based policy evaluation called the certifier Service. It simplifies and unifies programming and

Confidential Computing and the Certifier Framework

Verifiably secure operational properties, including confidentiality, integrity and policy compliance, no matter where program runs. Safe against malware and "insiders."

Before CC: developer/deployer must:

1. Write applications correctly
2. Deploy the program safely (no changes)
3. Configure 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

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

With CC: developer deployer must:

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

- Consequence: You can have safe application but it's platform dependent and a lot of work

With CC & Certifier: developer/deployer must:

1. Write the application correctly using Certifier APIs
2. Write the trust policy
3. Use Certifier Service to manage it!

- Consequence: You write the application once. Need only add a few dozen lines of code to enable CC protection. Trust policy is independent of application. Can move to another "backend" effortlessly

Thanks to David Wagner

Thank you!

John Manferdelli <jmanferdelli@vmware.com>

Overcoming Barriers to Confidential Computing as a Universal Platform

Abstract: Confidential Computing (CC) provides simple, principled confidentiality and integrity for workloads wherever they run. Within multi-cloud infrastructures, it opens the door for a universal distributed computing solution that addresses verifiable program isolation, programs as authenticated security principals, secure key management, trust management, and the ability to prove these security properties cryptographically “over the wire” to relying parties using attestation. Yet the adoption of confidential computing has been slowed by the difficulty of writing CC-enabled programs quickly and securely, and across hardware technologies. Manferdelli will describe issues and requirements for a universal programming platform and introduce the open source “Certifier Framework for Confidential Computing” that provides a step towards overcoming development barriers.