Certifier Framework for Confidential Computing
Proposal for inclusion as a Confidential Computing Consortium Project

Submitted by VMware CC-Certifier Framework development team, October, 2023

General Information

1. **Name of the Project**: Certifier Framework for Confidential Computing
2. **Project Description**:

   The Certifier Framework (CF) provides a platform-agnostic API to simplify the development and deployment of applications using Confidential Computing (CC) platforms.

   The Certifier Framework is an open-source software package comprising of:
   - Certifier API library: A client API, written in C++, used for CC application development., written in C++ and provides Python-language bindings.
   - Certifier Service: A server-based policy-evaluation server, written in Go-lang, to certify CC-based applications.
   - Sample programs developed using this API for multiple secure enclaves (h/w platforms)
   - Utility programs to manage policy, keys and certificates.

   The Certifier Framework supports a simulated enclave, AMD SEV-SNP, Intel SGX, RISC-V and Arm-CCA platforms.

3. **Alignment with Consortium's Mission Statement**: "*The Confidential Computing Consortium (CCC) brings together hardware vendors, cloud providers, and software developers to accelerate the adoption of Trusted Execution Environment (TEE) technologies and standards.*"

   The mission of the Certifier Framework Project is to simplify and unify application programming and operations support for multi-vendor Confidential Computing platforms by providing simple, scalable, and policy-driven trust management.

   The goal is to **simplify and standardize** the development of secure software, rooted in Confidential Computing and to enable the **scalable deployment** of such software on client-, cloud- and on-premise environments.

   The Certifier Framework provides abstracted interfaces to Confidential Computing primitives (such as Seal, Unseal, Attestation, Measurement and Verification) under an open, standardized and simple-but-comprehensive API across many Trusted Execution Environments (TEEs). In addition, the framework provides recursive TEE support (TEE-within-a-TEE). For example, the framework allows encrypted virtual-machine level TEEs (like AMD

SEV-SNP) to provide application-level (process) TEE-like services for applications that run in the encrypted virtual machine.

The Framework employs many of the Consortium's current projects, including Gramine, OpenEnclaves SDK and Keystone.

Consequently, the Certifier Framework's mission aligns and contributes to the foregoing stated mission of the Consortium.

4. **Project website**: https://github.com/vmware-research/certifier-framework-for-confidential-computing

5. **Social Media accounts**: (None)

Legal Information

1. **Project Logo**: (None)

2. **Project License**: Apache License 2.0 (See GitHub license.md).

    a. Some test drivers are licensed under GPL 2.0. There are very few GPL-affected files. None of the GPL-licensed code is required in the Certifier API, sample applications, utility programs or the Certifier Service.
    b. All other dependent software packages used as part of the build-process and run-time (e.g., OpenSSL, Google protocol buffers, Python, pytest, SWIG tool etc.) are open-source and used under their respective licenses.

3. **Existing financial sponsorship**: VMware Office of the CTO (OCTO)

4. **Trademark status**: Currently, there are no artifacts in the Certifier Framework repository and collateral which are trademarked.

5. **Proposed Technical Charter**: As mentioned above, the Certifier Framework allows developers and application deployers to develop and manage TEE's securely on a variety of platforms with little modification to existing secure software. (See attached CertifierFramework-Technical-Charter.pdf file.)

Technical Information

1. **High level assessment of project synergy** with existing projects under the CCC (*including how the project compliments / overlaps with existing projects, and potential ways to harmonize over time*.):

The Certifier Framework already uses, for the purposes outlined by the Consortium's mission, half the projects under the CCC projects umbrella. We use Gramine SDK, OpenEnclaves SDK and Keystone. In addition, the project uses and supports AMD's tools and is working with organizations that plan to submit additional standards, including Samsung. In addition, in the future, we may employ components from Occlum and Veraison to implement underlying functions.

2. **The Trusted Computing Base (TCB) of the project**:

   Protobufs, OpenSSL, standard C++ and Go libraries, SWIG-generated wrapper-C++ code, Python drivers. In addition, we rely on the correctness of underlying compilers and these libraries. Some enclaves also rely on the correctness of the incorporated software like Gramine, OpenEnclave, Keystone SDK and Islet (Samsung) SDK. For many of the enclaves, including encrypted virtual machines, there is an additional dependency on the correctness, depending on the specific use, of identified firmware.

3. **Project Code of Conduct URL**: https://github.com/vmware-research/certifier-framework-for-confidential-computing/blob/main/CODE_OF_CONDUCT.md

4. **Source control URL**: https://github.com/vmware-research/certifier-framework-for-confidential-computing

5. **Issue tracker URL**: https://github.com/vmware-research/certifier-framework-for-confidential-computing/issues

6. **External dependencies**:

   Protobufs, gtest, gflags, OpenEnclaves and Gramine SDKs and standard C++ and Go libraries (listed below). Our CI build.yml installs all these software packages: clang-format-11 libgtest-dev libgflags-dev openssl libssl-dev protobuf-compiler protoc-gen-go golang-go cmake.

   We also used some vendor provided open-source tools to provide program measurement (e.g., AMD virtee tools).

   The Certifier Framework relies on the following Open-Source software packages and libraries:

Package / Library	Dependency at	
	Build-time	Run-time
OpenSSL	✓	✓
Google Protocol Buffers	✓	✓
Go-lang Go, protoc-gen-go	✓	✓
Google gflags	✓	✓

Google gtest	✓	✓
SWIG	✓	
Python3, pytest		✓
g++, clang-format, cmake	✓	

7. **Standards implemented by the project**:
    - IETF-RATS: https://datatracker.ietf.org/doc/draft-ietf-rats-eat/

8. **Release methodology and mechanics**: Currently, the Certifier Framework is in an early-beta stage in a publicly accessible Git repository. We expect a stable "v.9" release around October 2023.

9. **Names of initial committers**: (Also see: MAINTAINERS.md)
    - John Manferdelli
    - Ye Li
    - Aditya Gurajada
    - Radoslav Gerganov

    External Collaborators:
    Samsung (2)
    AMD (1)
    RISC-V (UC Berkeley) (1)
    Anyscale (1)

10. **List of project's official communication channels**: GitHub Discussions is our current channel.

11. **Project Security Response Policy**:

    On Git website, see SECURITY.MD (which is similar to the security policy adopted by Veracruz, security.md). The security policy process will be enforced starting with our upcoming v0.9 release.

12. **Preferred maturity level**: Incubation

13. **Additional information**:

    - Certifier Framework Whitepaper: RepositoryWhitepaper.pdf

    - 3-part blog on VMware Open-Source blogs site:

        a. https://blogs.vmware.com/opensource/2022/11/22/confidential-computing-part-1-tackling-challenges/

b. https://blogs.vmware.com/opensource/2022/12/01/confidential-computing-part-2-the-technical-bits/

c. https://blogs.vmware.com/opensource/2022/12/08/confidential-computing-part-3-the-certifier-framework/

- CCC-Summit, '23 talks: 2023-CC-SummitKeynote-Final.pdf (pptx), 2023-CC-SummitTechnical-Final.pdf (pptx)

- Certifier Framework documentation: Doc, sample-apps-README.md,

Glossary

- CC: Confidential Computing
- CF: Certifier Framework
- TCB: Trusted Computing Base
- TEE: Trusted Execution Environment