# Portable Programing Environment and API on Trusted Execution Environment (TEE)

# User's Manual

2024-12-04

# 1 Overview of Trusted Application Reference (TA-Ref)

There is a wiki page describing Introduction, objective and use cases of TEEP Protoc ol.

- https://github.com/ietf-teep/teep-protocol/wiki

The TEE is a feature of having capability of running software from an isolated area assisted by CPU hardware.

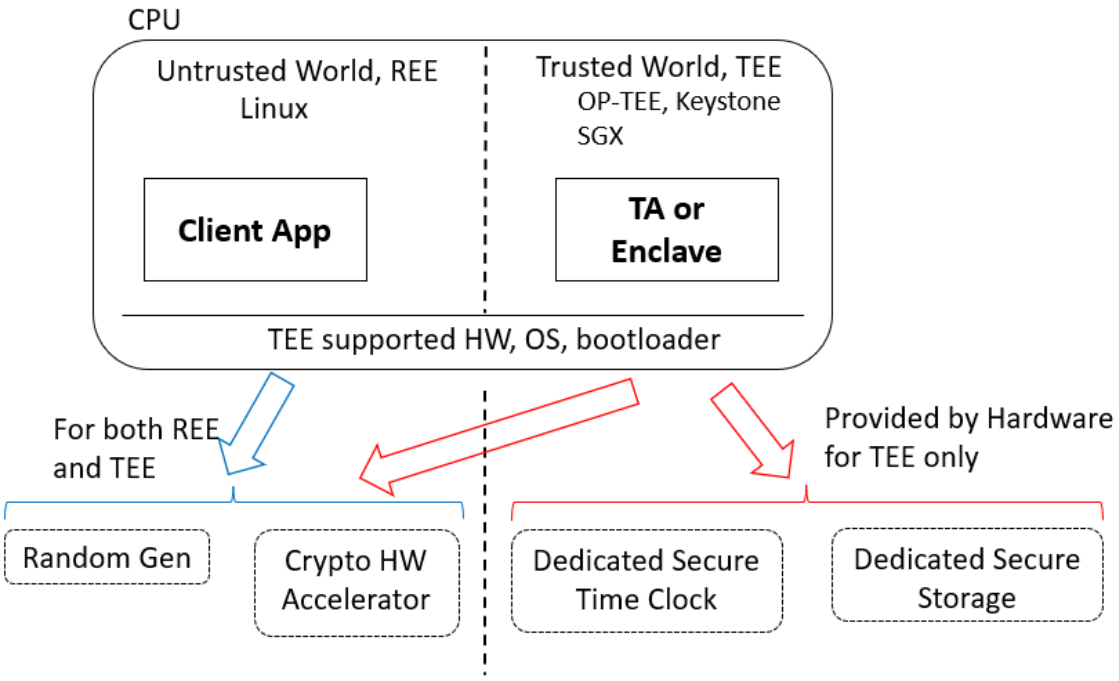
Many recent devices are able to be customized by installing softwares from end users, dealers and security service companies which are different entities from the device vendors, for example, smart phones, Android TVs, set top boxes. infotainment system on cars, surveillance cameras, home security gateways, edge routers, network equipment, and etc. In this situation, malicious software could be installed on a regular operating system, such as Linux.

The TEE provides a secure software runtime environment for the security sensitive software from preventing interference on customizable devices of softwares running on a normal operating system.

## 1.1 Features of TA-Ref

- Provides Portable API and SDK among Intel SGX, ARM TrustZone-A and RISC-V Keystone
- Provides portability for source codes of Trusted Applications among SGX, TrustZone and Keystone
- Provides subset of Global Platform API on TEE
- Tutorial programs of common usage of hash functions, symmetric algorithm and asymmetric algorithm
- Simple Makefiles to develop TAs on different CPUs which traditionally tend to have complex build systems

### 1.1.1 Assumption of hardware features of TA-Ref on TEE




The Secure Time Clock is the date and time clock hardware peripheral which updates monotonically provided separately from regular clock peripheral so the user application and OS on REE could not change the date and/or time. Many certificates of CA, license keys of purchased serial code, hardware enablement keys such as increasing the battery size of the electric cars are bound to the date. The easiest way for end users or attackers phishing the CAs and web sites, using the software and enabling the optional hardware feature without the payment is to change the value of the clock. The concrete date and time is especially important for the telemetry data.

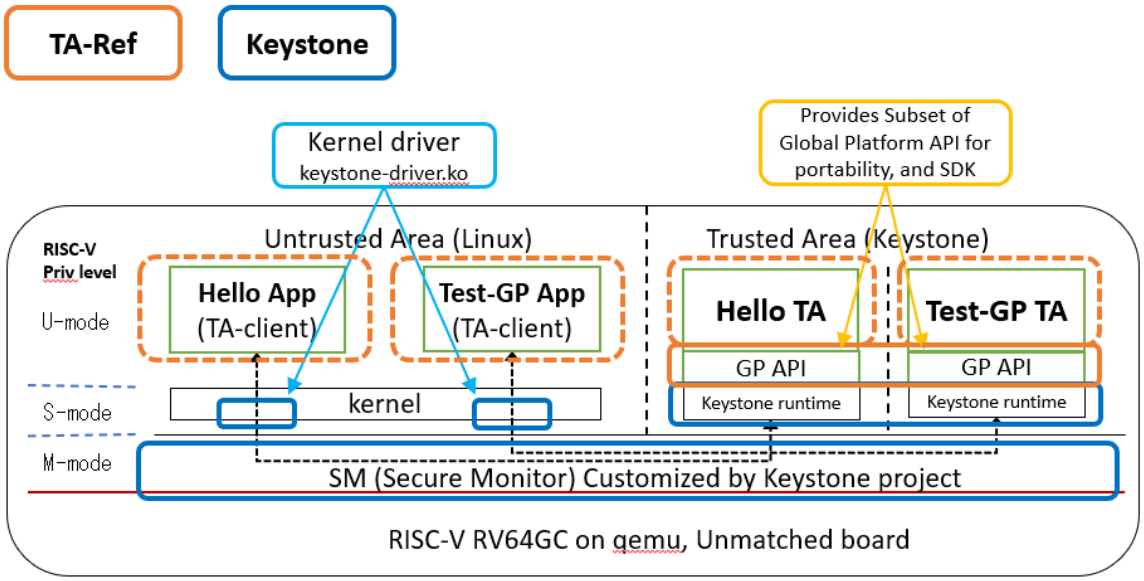
The Secure Storage will be saving the cryptographic keys, Trusted Application binaries, personalization data, telemetry data, and etc, which are security sensitive files must not be tampered by any applications on the REE side. The size of the storage is typically in the order of megabytes to fulfill the required files.

The Random Generator is another requirement of the hardware ensuring the security level of the system. Quality of the random value is very important for having a good security level on many cryptographic algorithms used inside TEE. It is recommended to have an equivalent level of SP 800-90B and FIPS 140-3.

The Cryptographic Hardware accelerators are not strictly mandatory hardware features, however, it is essential to have them to be usable devices to prevent or expose the very slow usability.

## 1.2 Components of TA-Ref

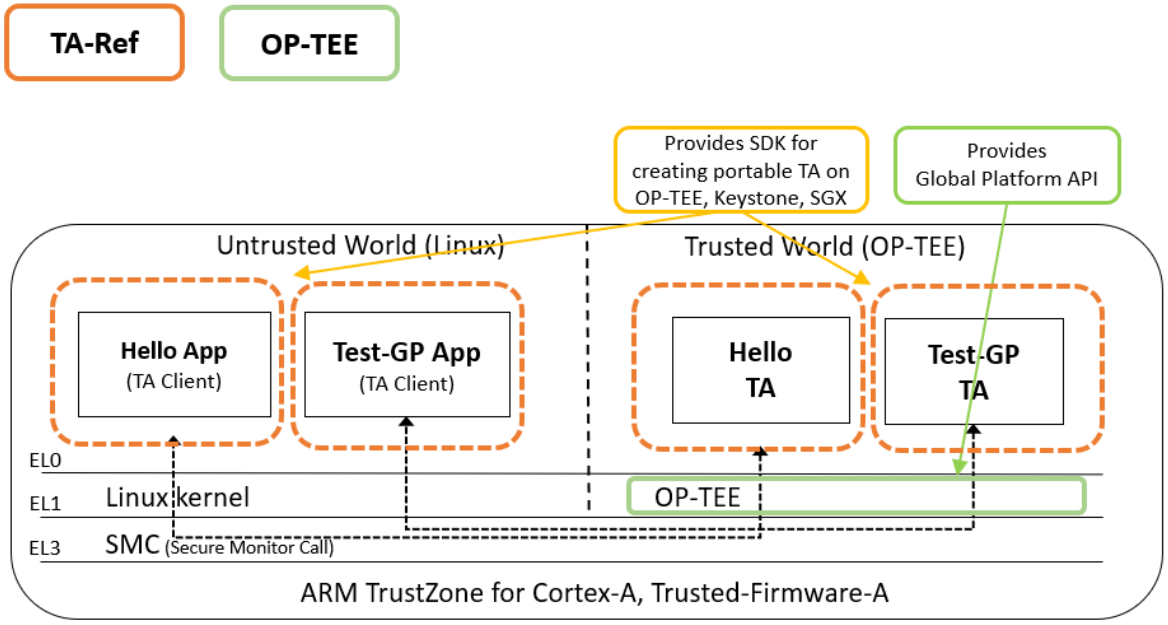
### 1.2.1 TA-Ref Components on Keystone




TA-Ref provides a portable TEE programming environment over the Keystone project on RISC-V RV64GC CPU. Each TA in the Trusted Aria is protected with Physical memory protection (PMP) which is enabled by RISC-V hardware.

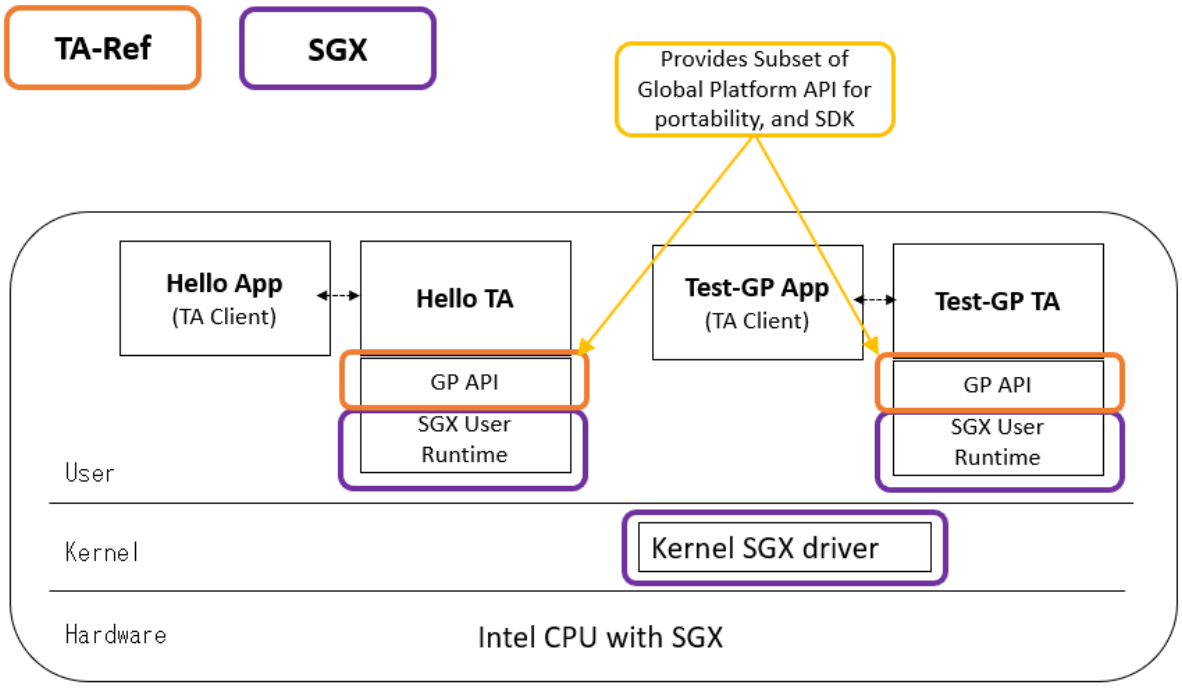
- Keystone project
  - https://keystone-enclave.org/

### 1.2.2 TA-Ref Components on OP-TEE




It is on OP-TEE and highly utilizing the programming environment provided by TA-Ref to simplify developing of Trusted Applications to be able to build and function on other CPUs with the single source code.

### 1.2.3 TA-Ref Components on SGX




The diagram shows implementation of TA-Ref and Trusted Applications on SGX. Unlike ARM Cortex-A or RISC-V, the TEE security level is implemented vertically in the user space. The TA-Ref provides the same programming environment of ARM Cortex-A or RISC-V on Intel with the capability subset of Global Platform TEE Internal APIs.

## 1.3 What we did on RISC-V

- We designed the GP internal API library to be portable.
- Keystone SDK is utilized because of runtime "Eyrie".
- The library is ported to Intel SGX as well as RISC-V Keystone.

### 1.3.1 Challenges faced during Implementation

- The combination of GP internal API and cipher suite is big.
  - To reduce the size, We pick up some important GP internal APIs.
- Some APIs depend on CPU architecture.
  - We separate APIs into CPU architecture dependent / independent.
- Integrate GP TEE Internal API to Keystone SDK.
  - Keystone SDK includes EDL (Enclave Definition Language) named “keedger”.
  - Keedger creates the code for OCALL (request from TEE to REE) to check the pointer and boundary.

### 1.3.2 Selected GP TEE Internal API's for testing

- CPU architecture dependent
  - Random Generator, Time, Secure Storage, Transient Object(TEE_GenerateKey)
- CPU architecture independent(Crypto)
  - Transient Object(exclude TEE_GenerateKey), Crypto Common, Authenticated Encryption, Symmetric/Asymmetric Cipher, Message Digest

Following shows the table of CPU Dependent and Independent API's with its functions.

| Category | CPU (In)Dependent | Functions |
|---|---|---|
| Random Number | Dependent | TEE_GenerateRandom |
| Time | Dependent | TEE_GetREETime, TEE_GetSystemTime |
| Secure Storage | Dependent | TEE_CreatePersistentObject, TEE_OpenPersistentObject, TEE_ReadObjectData, TEE_WriteObjectData, TEE_CloseObject |
| Transient Object | Dependent Independent | TEE_GenerateKey, TEE_AllocateTransientObject, TEE_FreeTransientObject, TEE_InitRefAttribute, TEE_InitValueAttribute, TEE_SetOperationKey |
| Crypto Common | Independent | TEE_AllocateOperation, TEE_FreeOperation |
| Authenticated Encryption | Independent | TEE_AEInit, TEE_AEUpdateAAD, TEE_AEUpdate, TEE_AEEncryptFinal, TEE_AEDecryptFinal |
| Symmetric Cipher | Independent | TEE_CipherInit, TEE_CipherUpdate, TEE_CipherDoFinal |
| Asymmetric Cipher | Independent | TEE_AsymmetricSignDigest, TEE_AsymmetricVerifyDigest |
| Message Digest | Independent | TEE_DigestUpdate, TEE_DigestDoFinal |

## 1.4 Dependency of category