INSTITUTE of INFORMATION SECURITY

Remote Attestation on Arm TrustZone OP-TEE with VERAISON Verifier --- current status and future plan ---

FOSDEM 2025 Attestation Devroom
Feb/2/2025 @Burussels

Kuniyasu Suzaki
IISEC: Institute of Information Security, Graduate School
This work is collaborated with Yuichi Sugiyama@Ricerca Security

https://github.com/iisec-suzaki/optee-ra

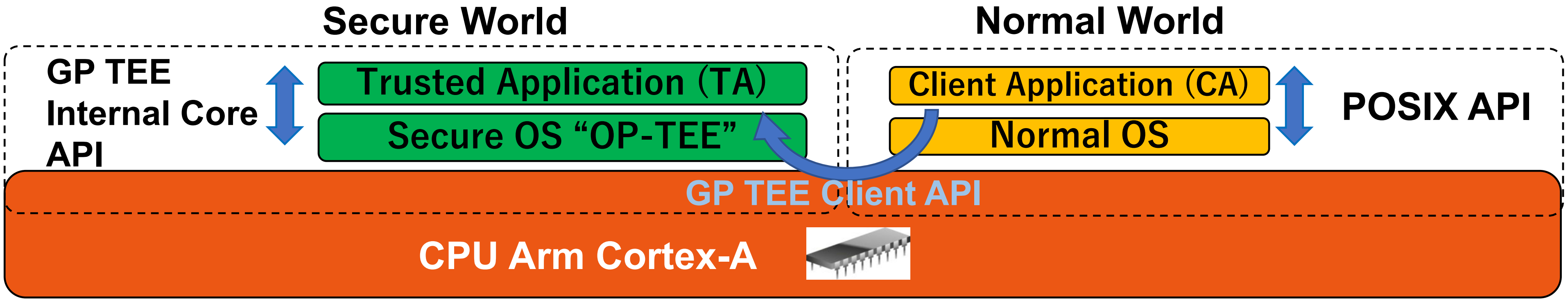
Contents

- What is OP-TEE on Arm TrustZone?
- What is VERAISON Verifier?
- Remote Attestation OP-TEE with VERAISON Verifier
  - Prerequisite
  - Provision Phase
  - Remote Attestation Phase
- Current Status
- Future Plan
  - Key Management using HSM (Hardware Security Module)
  - Secure Boot Confirmation
  - Certificate-Based Attestation Keys
- Conclusions

What is OP-TEE on Arm TrustZone?

https://github.com/OP-TEE/optee_os

- Arm Cortex-A TrustZone is a popular TEE on smartphones.
- OP-TEE is an open source Secure OS for Arm Cortex-A TrustZone, which follows the API specifications of GlobalPlatform.
- OP-TEE had a simple attestation mechanism but it does not satisfy current remote attestation.
- We developed the total remote attestation for OP-TEE.



What is VERAISON Verifier

https://github.com/veraison

VERAISON

INSTITUTE of INFORMATION SECURITY

- VERAISON is an open source verifier based on IETF RATS (Remote ATtestation proceduеS).

⬭ : Roles ⟶ : Artifact

- 2 Phase
  - Provisioning
  - Remote Attestation

- RATS requires CBOR formant for attestation evidence.

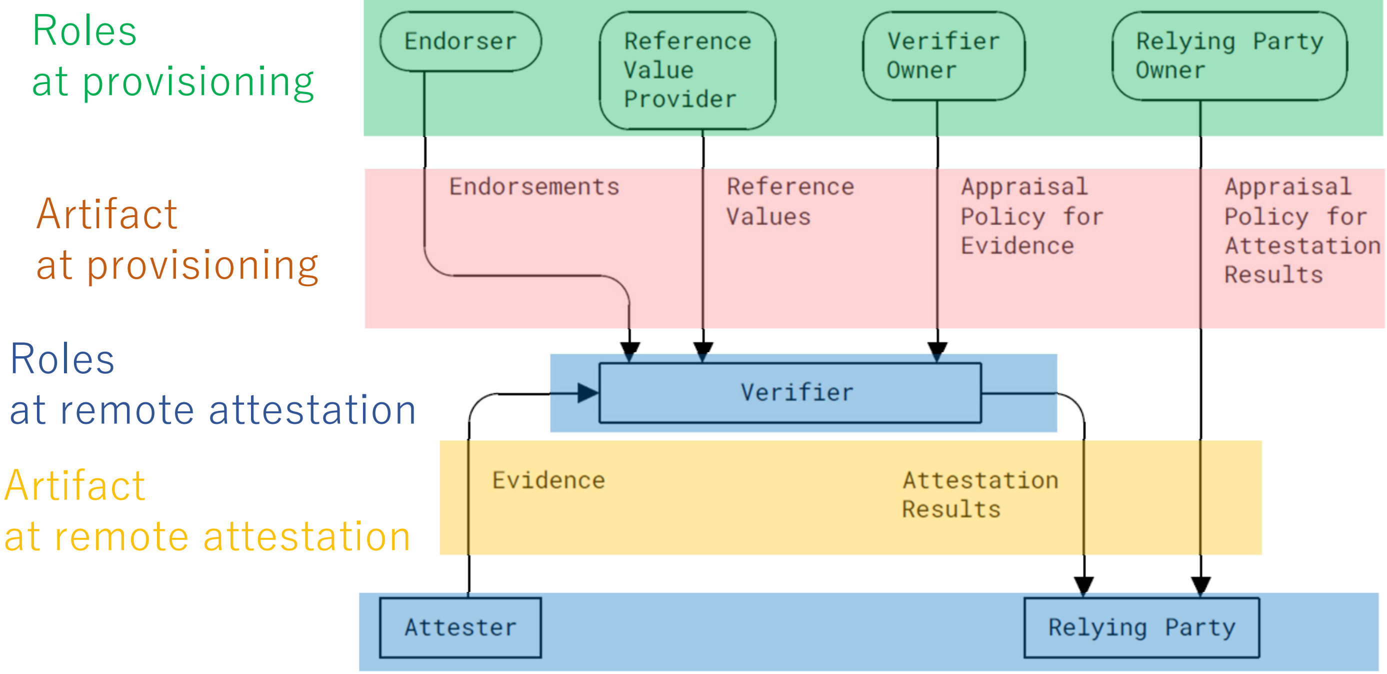
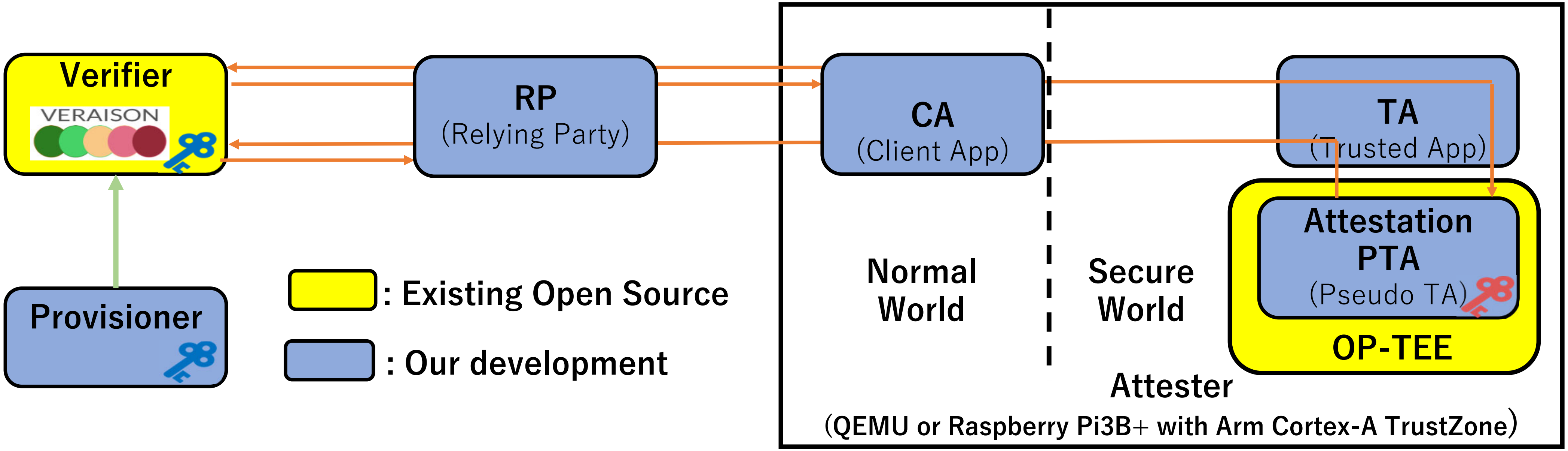


Figure 1: Conceptual Data Flow

What we developed

- On OP-TEE
  - OP-TEE's API to create Attestation Evidence of TA.
  - PTA (Pseudo TA)
    - Measure the SHA-256 of TA
    - Create an Attestation Evidence with CBOR format
    - Sign the Attestation Evidence with ECDSA w/ SHA-256

- Sample TA
- Sample CA
- Sample Relying Party
- Sample setting of VERAISON



Verifier

VERAISON

RP
(Relying Party)

CA
(Client App)

TA
(Trusted App)

Attestation PTA
(Pseudo TA)

OP-TEE

Normal World

Secure World

Provisioner

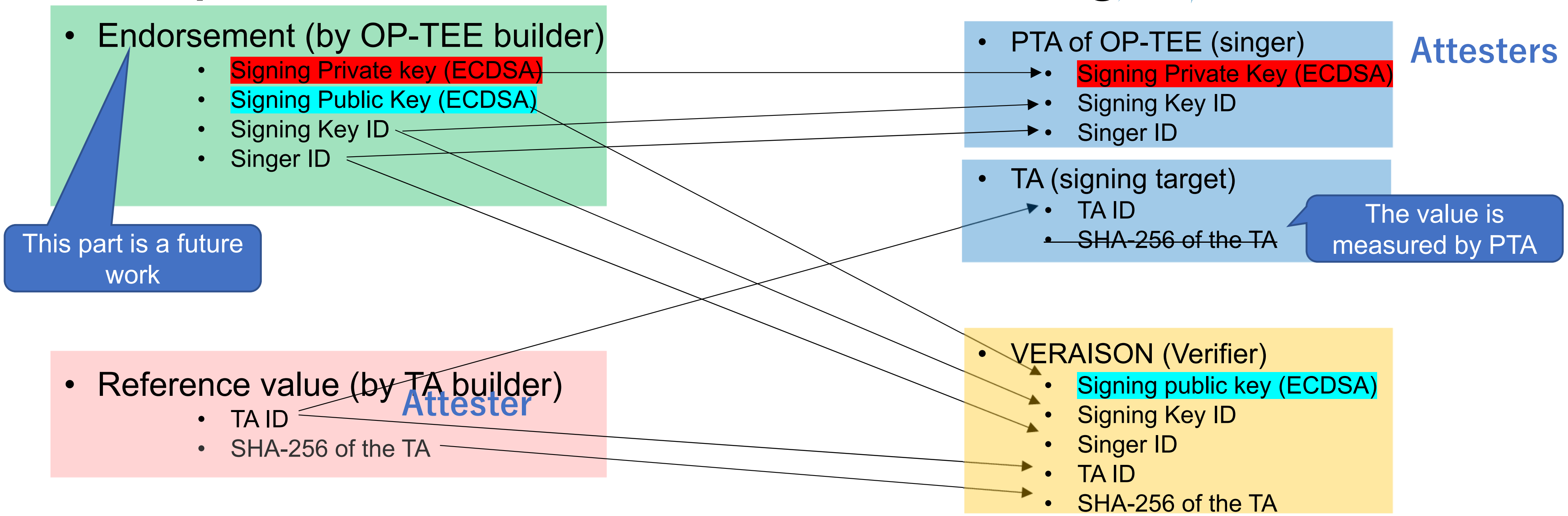
🟨 : Existing Open Source

🟦 : Our development

Attester
(QEMU or Raspberry Pi3B+ with Arm Cortex-A TrustZone)

Prerequisite

Provisioning



- Endorsement (by OP-TEE builder)
  - Signing Private key (ECDSA)
  - Signing Public Key (ECDSA)
  - Signing Key ID
  - Singer ID

This part is a future work

- Reference value (by TA builder)
  - TA ID
  - SHA-256 of the TA

Attester

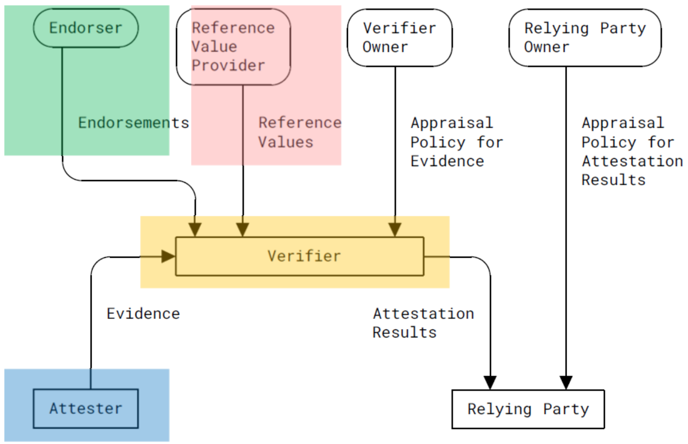
- PTA of OP-TEE (singer)
  - Signing Private Key (ECDSA)
  - Signing Key ID
  - Singer ID

Attesters

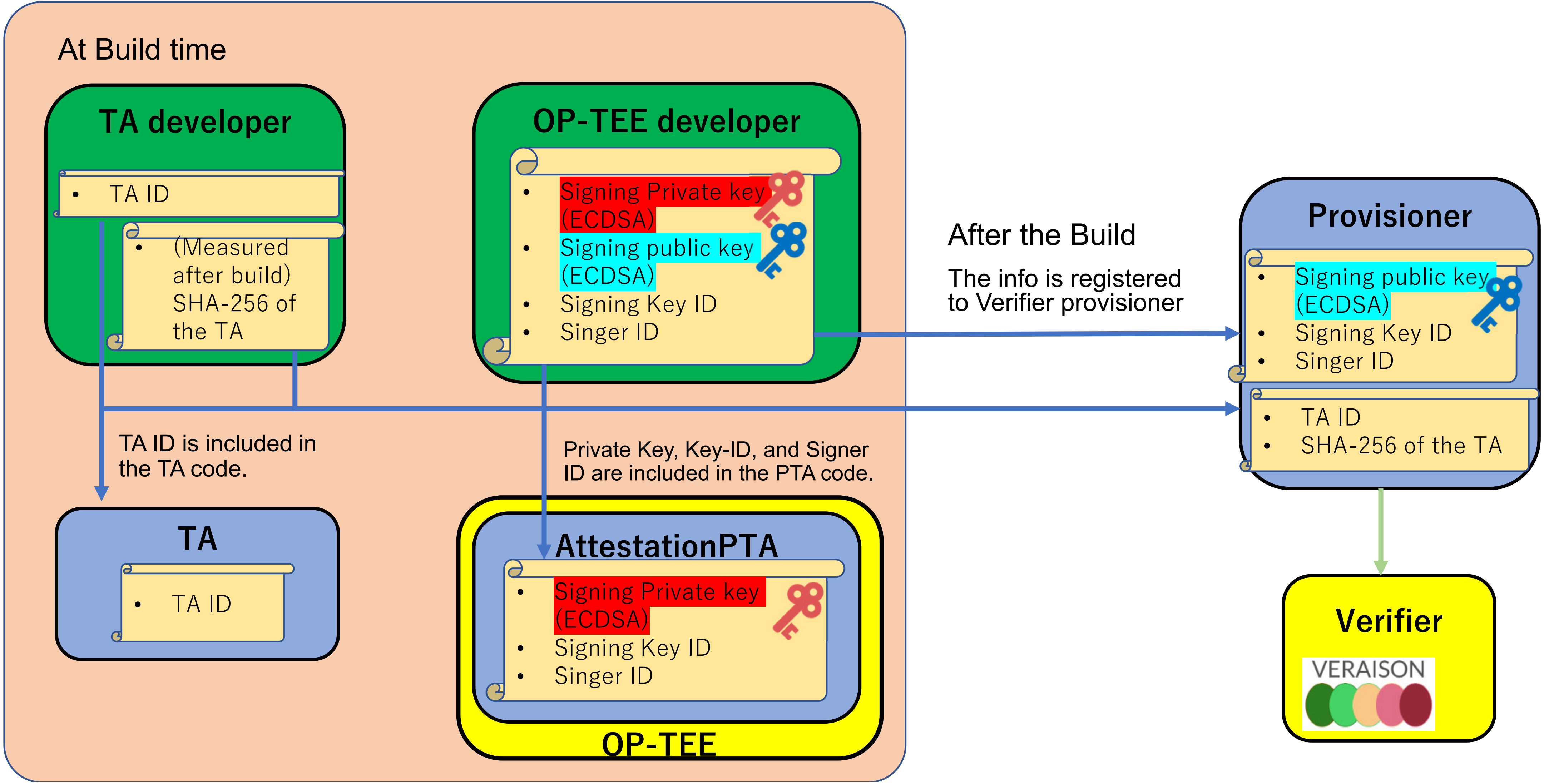
- TA (signing target)
  - TA ID
  - ~~SHA-256 of the TA~~

The value is measured by PTA

- VERAISON (Verifier)
  - Signing public key (ECDSA)
  - Signing Key ID
  - Singer ID
  - TA ID
  - SHA-256 of the TA



Provisioning



At Build time

TA developer

- TA ID
- (Measured after build) SHA-256 of the TA

OP-TEE developer

- Signing Private key (ECDSA)
- Signing public key (ECDSA)
- Signing Key ID
- Singer ID

TA ID is included in the TA code.

Private Key, Key-ID, and Signer ID are included in the PTA code.

TA

- TA ID

AttestationPTA

- Signing Private key (ECDSA)
- Signing Key ID
- Singer ID

OP-TEE

After the Build

The info is registered to Verifier provisioner

Provisioner

- Signing public key (ECDSA)
- Signing Key ID
- Singer ID

- TA ID
- SHA-256 of the TA

Verifier

VERAISON

VERAISON Provisioning

```
TRUST ANCHORS:
--------------
{
  ”scheme”: ”PSA_IOT”,
  ”type”: ”trust anchor”,
  ”subType”: ””,
  ”attributes”: {
    ”PSA_IOT.hw-model”: ”RoadRunner”,
    ”PSA_IOT.hw-vendor”: ”ACME”,
    ”PSA_IOT.iak-pub”: ”-----BEGIN PUBLIC KEY-----
¥nMFkwEwYHKoZIzjOCAQYIKoZIzjODAQcDQgAEMKBCTNIcKUSDii11ySs3526iDZ8A¥niTo7Tu6KPAqv7D7gS2XpJFbZiItSs3m9+9Ue6GnvHw/GW2ZZaVtszggXIw==¥n-----END PUBLIC KEY-----”,
    ”PSA_IOT.impl-id”: ”YWNtZS1pbXBsZW1lbnRhdGlvbi1pZCOwMDAwMDAwMDE=”,
    ”PSA_IOT.inst-id”: ”Ac7rrnuJJ6MiflMDz14PH3sOu1Qq1yUKwD+83jbsLxUI”
  }
}
```

Signing public key (ECDSA)

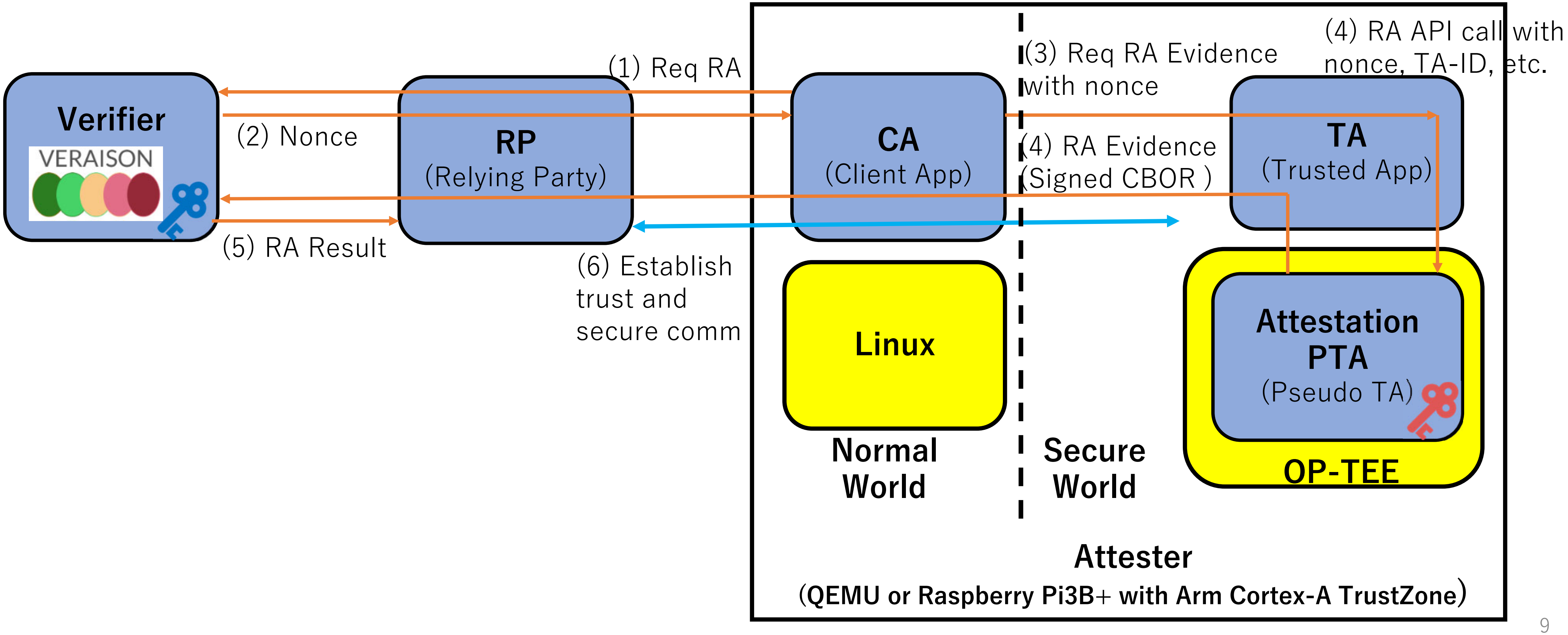
```
ENDORSEMENTS:
-------------
{
  ”scheme”: ”PSA_IOT”,
  ”type”: ”reference value”,
  ”subType”: ”PSA_IOT.sw-component”,
  ”attributes”: {
    ”PSA_IOT.hw-model”: ”RoadRunner”,
    ”PSA_IOT.hw-vendor”: ”ACME”,
    ”PSA_IOT.impl-id”: ”YWNtZS1pbXBsZW1lbnRhdGlvbi1pZCOwMDAwMDAwMDE=”,
    ”PSA_IOT.measurement-desc”: ”sha-256”,
    ”PSA_IOT.measurement-type”: ”PRoT”,
    ”PSA_IOT.measurement-value”: ”MbgFqjT4jfR+fK104YyQtZUYDOnhXh7GfhMOEmR6tgc=”,
    ”PSA_IOT.signer-id”: ”rLsRx+TaIXIFUjzkzhokWuGiOa48a/2eeHH35di66Gs=”
  }
}
```

TA ID

SHA-256 of the TA

Singer ID

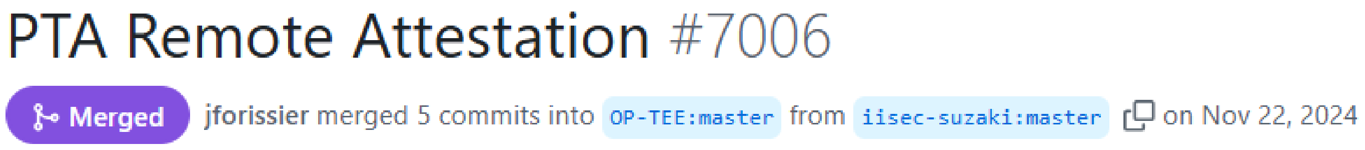
Remote Attestation Phase



Verifier
VERAISON
RP
(Relying Party)
(1) Req RA
(2) Nonce
(5) RA Result
(6) Establish trust and secure comm
CA
(Client App)
(3) Req RA Evidence with nonce
(4) RA Evidence (Signed CBOR)
(4) RA API call with nonce, TA-ID, etc.
TA
(Trusted App)
Linux
Attestation PTA
(Pseudo TA)
OP-TEE
Normal World
Secure World
Attester
(QEMU or Raspberry Pi3B+ with Arm Cortex-A TrustZone)

Current status

- The code for OP-TEE was merged on Nov 22, 2024.
  - https://github.com/OP-TEE/optee_os/pull/7006



PTA Remote Attestation #7006

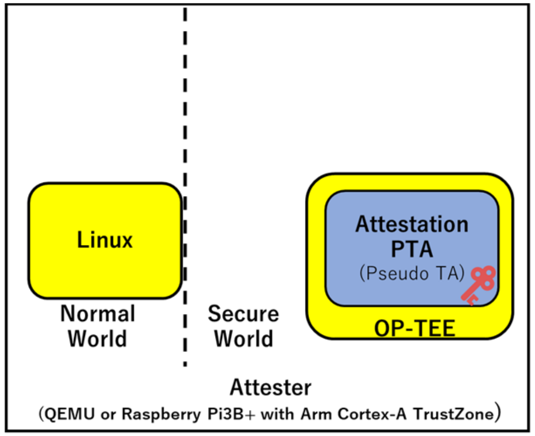
Merged jforissier merged 5 commits into OP-TEE:master from iisec-suzaki:master on Nov 22, 2024

- The samples are confirmed on QEMU and Raspberry Pi3 B+
  - https://github.com/iisec-suzaki/optee-ra

- We are now trying to add the samples to OP-TEE Examples
  - https://github.com/linaro-swg/optee_examples

Future Plan
(1) Key Management using HSM (Hardware Security Module)

- Current implementation embeds the signing private key in the PTA binary.
  - Attacker can get the key from the boot storage.


Linux
Normal World
Secure World
Attestation PTA (Pseudo TA)
OP-TEE
Attester
(QEMU or Raspberry Pi3B+ with Arm Cortex-A TrustZone)

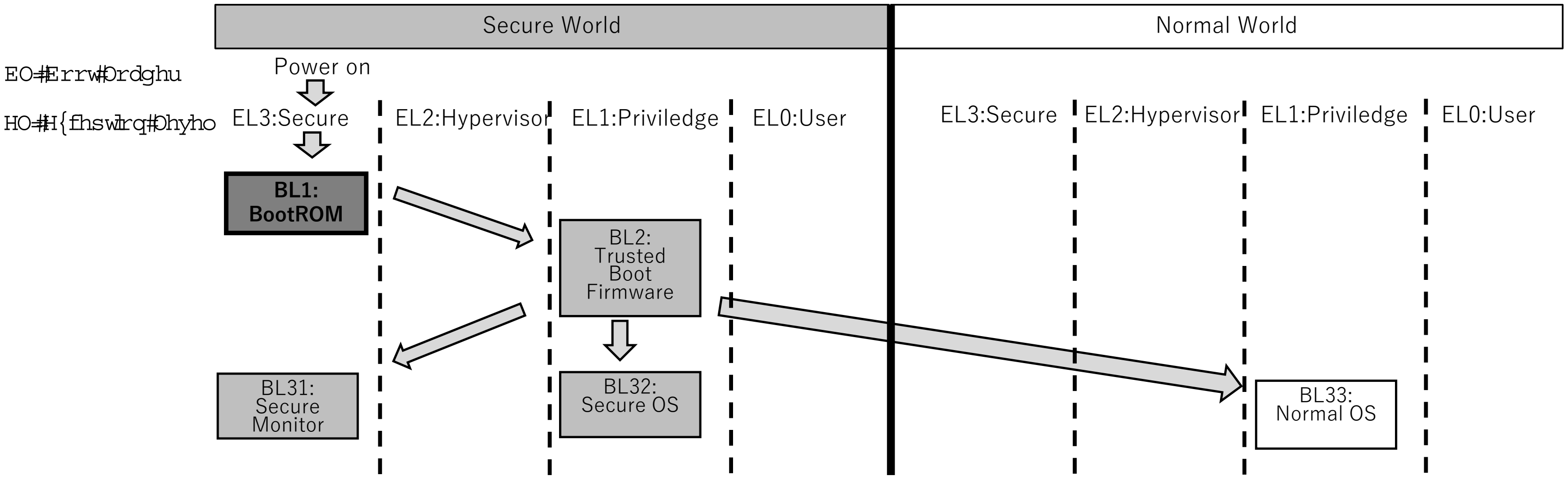
- Solution
  - Whitebox Cryptography
  - HSM (Hardware Security Module)

- Our approach
  - HSM based on SE (Secure Element) or CAAM (Cryptographic Accelerator and Assurance Module) of NXP

Future Plan

(2) Secure Boot Confirmation

- The boot process of Arm TrustZone is mutable and vulnerable for root-kit attacks.



- Remote attestation needs to confirm the secure boot of Secure OS (i.e. OP-TEE).

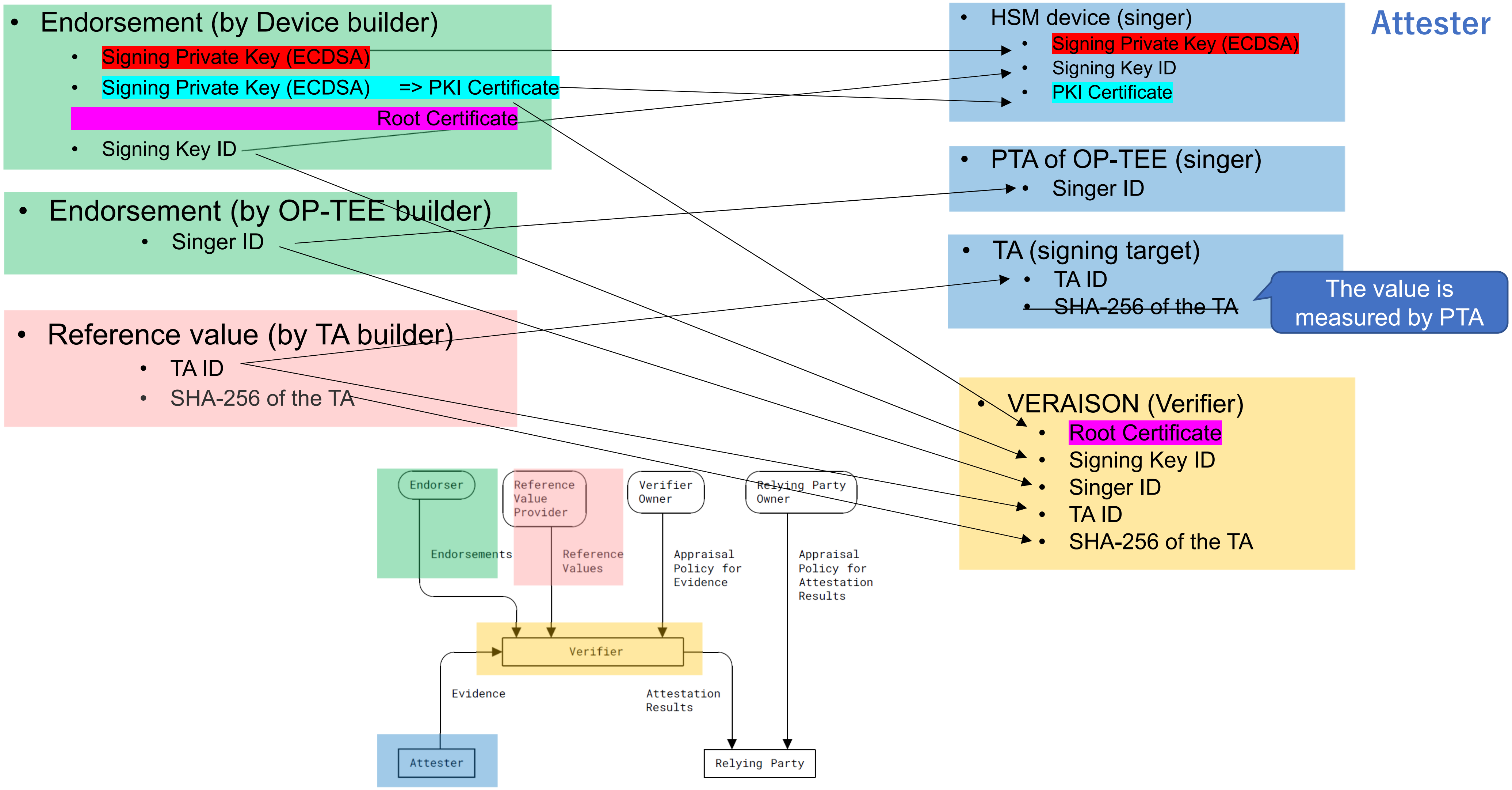
Future Plan

(3) Certificate-Based Attestation Keys

- Current implementation uses signing keys directly.
  - Each device has its own private key, and the verifier must have all public keys.
  - This model is not scalable!
- PKI Certificate based Attestation Keys
  - Device builder becomes an Endorsement.
  - Each device has its own key’s PKI certificate, and the verifier has the issuer’s root certificate.
  - Pros
    - Scalability
  - Cons
    - The Endorser must take a PKI certificate for a signing key.
    - The vesication process is a little bit complicated.

Prerequisite

Provisioning



- Endorsement (by Device builder)
  - Signing Private Key (ECDSA)
  - Signing Private Key (ECDSA) => PKI Certificate

    Root Certificate
  - Signing Key ID
- Endorsement (by OP-TEE builder)
  - Singer ID
- Reference value (by TA builder)
  - TA ID
  - SHA-256 of the TA

Attester

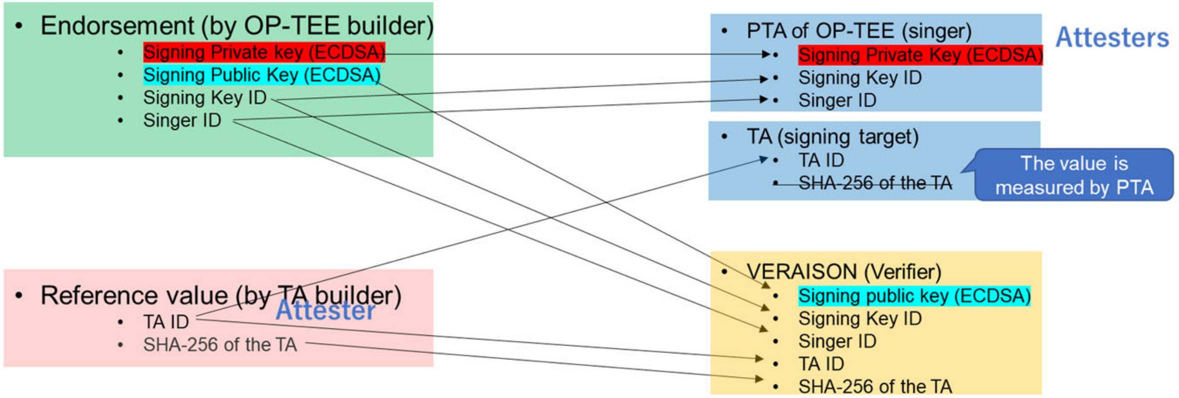
- HSM device (singer)
  - Signing Private Key (ECDSA)
  - Signing Key ID
  - PKI Certificate
- PTA of OP-TEE (singer)
  - Singer ID
- TA (signing target)
  - TA ID
  - ~~SHA-256 of the TA~~

The value is measured by PTA

- VERAISON (Verifier)
  - Root Certificate
  - Signing Key ID
  - Singer ID
  - TA ID
  - SHA-256 of the TA

Comparison (current and future)

- Current



- Endorsement (by OP-TEE builder)
  - Signing Private key (ECDSA)
  - Signing Public Key (ECDSA)
  - Signing Key ID
  - Singer ID
- Reference value (by TA builder)
  - TA ID
  - SHA-256 of the TA

Attester

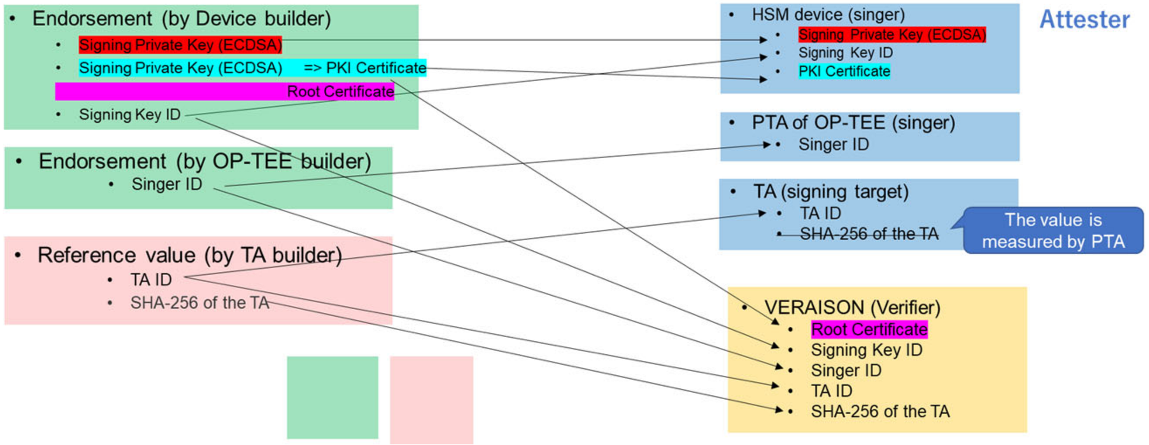
- PTA of OP-TEE (singer)
  - Signing Private Key (ECDSA)
  - Signing Key ID
  - Singer ID
- TA (signing target)
  - TA ID
  - SHA-256 of the TA

Attesters

The value is measured by PTA

- VERAISON (Verifier)
  - Signing public key (ECDSA)
  - Signing Key ID
  - Singer ID
  - TA ID
  - SHA-256 of the TA

- Future



- Endorsement (by Device builder)
  - Signing Private Key (ECDSA)
  - Signing Private Key (ECDSA) => PKI Certificate
  - Root Certificate
  - Signing Key ID
- Endorsement (by OP-TEE builder)
  - Singer ID
- Reference value (by TA builder)
  - TA ID
  - SHA-256 of the TA

- HSM device (singer)
  - Signing Private Key (ECDSA)
  - Signing Key ID
  - PKI Certificate
- PTA of OP-TEE (singer)
  - Singer ID
- TA (signing target)
  - TA ID
  - SHA-256 of the TA

Attester

The value is measured by PTA

- VERAISON (Verifier)
  - Root Certificate
  - Signing Key ID
  - Singer ID
  - TA ID
  - SHA-256 of the TA

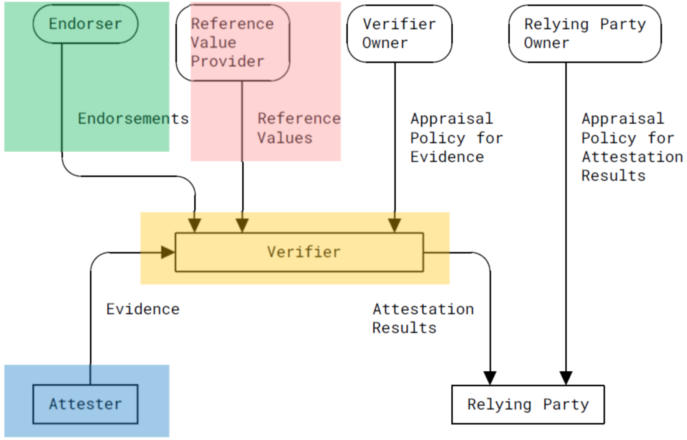


Figure 1: Conceptual Data Flow

This figure does not show improvement in scalability.

Discussion

- Does the Verifier need a SINGER-ID?
- If so, the code of SINGNER (OP-TEE PTA) (i.e., SHA-246 of OP-TEE) should be verified.

- The future plan’s Secure Boot Confirmation will be the answer.

Conclusion

- Report the OP-TEE with VERAISON Verifier
- current status
  - The code is included in original OP-TEE. It works but needs more security.
- future plan
  1. The key is protected by HSM.
  2. The evidence of secure boot is included in Attestation Evidence.
  3. The PKI Certificate is used for scalabilty.

Acknowledgment:
This work was supported by JST, the establishment of university fellowships towards the creation of science technology innovation, Grant Number JPMJFS2136.