TEEP (Trusted Execution Environment Provisioning) implementation on RISC-V

FOSDEM, 6 February 2021

Hardware-Aided Trusted Computing devroom

Akira Tsukamoto[1], Kuniyasu Suzaki[2,1], Kohei Isobe[2,3], Ken Takayama[3],
Masashi Kikuchi[2], Takahiko Nagata[2]

(1) National Institute of Advanced Industrial Science and Technology (AIST)

(2) Technology Research Association of Secure IoT Edge Application Based on RISC-V Open Architecture (TRASIO)

3) SECOM CO., LTD

@FOSDEM 2021

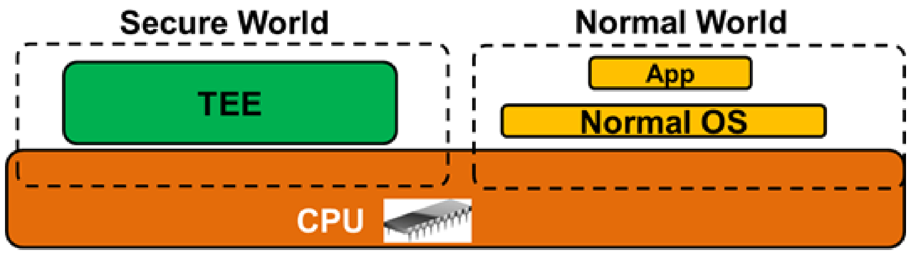
Agenda

- Introduction of TEE and Trusted Application (TA) Programming
- TEE on RISC-V
- Overview of TEEP at IETF
- TEEP on ARM Cortex-A (Initial Prototype)
- TEEP on RISC-V (under developing, porting from ARM)
- Recent activity of TEEP at IETF
- TEEP message examples
- Summary

Introduction of TEE

- Current OS and Hardware have many vulnerabilities, and Critical Applications are involved. Critical Applications are desired to be run independent from the OS.

- Trusted Execution Environment (TEE) is new CPU mechanism to offer “Secure World” which is isolated from the normal OS.
  - Critical Application is called “Trusted Application (TA)" or "Enclave".



- Popular CPU architectures provide TEE hardware
  - Intel SGX, AMD SEV, ARM TrustZone
  - RISC-V has PMP as TEE hardware

TEE consists of both hardware and software support

- TEE
  - Hardware-assisted Isolated Execution Environments
    - Provides processes to run at hidden partition from Regular OS

✚

- TEE Software Development Kit
  - Provides programming environment inside Isolated Execution Environments

Critical Application = Security sensitive operations or operate on sensitive data

- Payment, DRM, Authentication and etc

TEE runs Trusted Applcations (TA) in Isolated Execution Environments

TEE on RISC-V

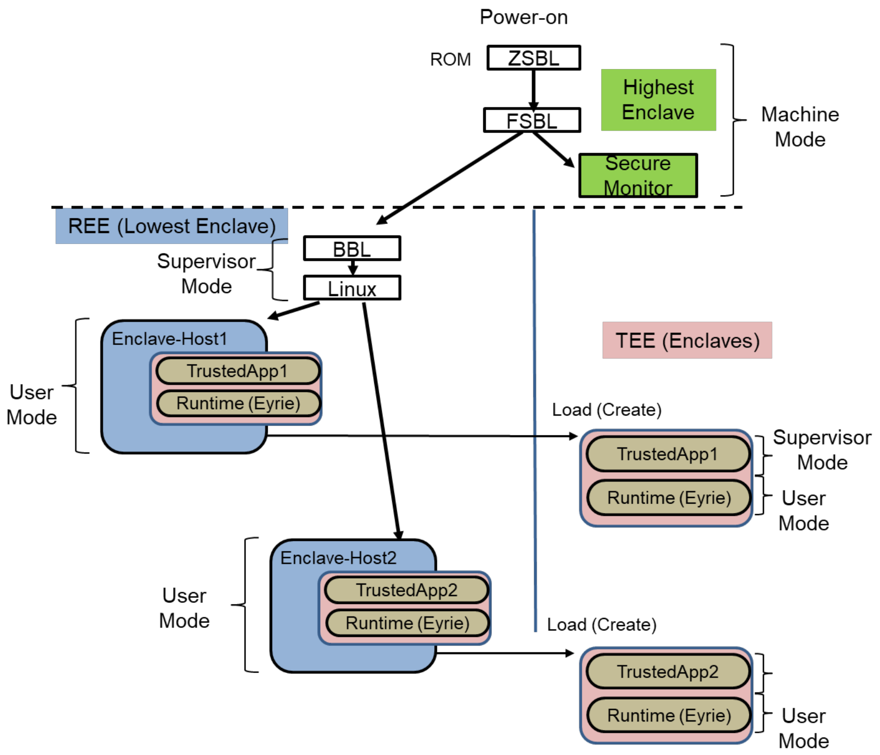
- RISC-V has some implementations of TEE.
  - MultiZone [HexFive]
  - Sanctum [MIT,USENIX Sec’16]
  - TIMBER-V [Graz University of Technology, NDSS’19]
  - MI6 [MIT,MICRO’19]
  - Keystone [UCB, EuroSys’20]

Reasons of choosing Keystone in our project

- Open source project, very active development
- Uses MMU
- Modular design to add our own features

Keystone project on RISC-V

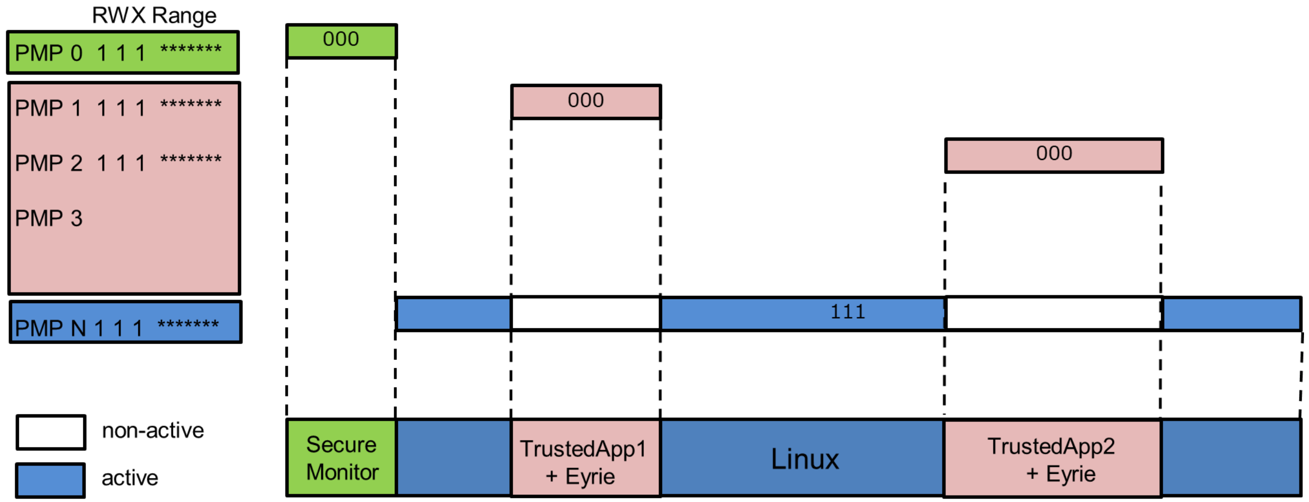
Keystone provides creation of Enclave (TA)



Boot procedure and Enclave (TA) creation

https://keystone-enclave.org/

RISC-V PMP provides Isolated Execution



Memory Management by PMP (Physical Memory Protection)

This figure shows partitioning Linux and Enclave

Use cases of Trusted Applications

- Targeted Devices
  - Smartphone, IoT, and Edge devices. (NAS, Edge Router, WIFI Router, Automotive Infotainment unit, Set-top box, Surveillance camera, Multifunction Printers and etc.)
  - Cloud Servers running Guest OSs.

- Payment, DRM, Authentication
  - e.g. Credit card app, PayPal, NetFlix, Cable TV, Mobile operator, Automotive, Insurance, etc.
- Secure firmware update
  - Injecting firmware as part of Trusted Application from TAM server.
- Confidential Cloud Computing
  - Prevent Host OS accessing User Data and Apps inside Guest OS.

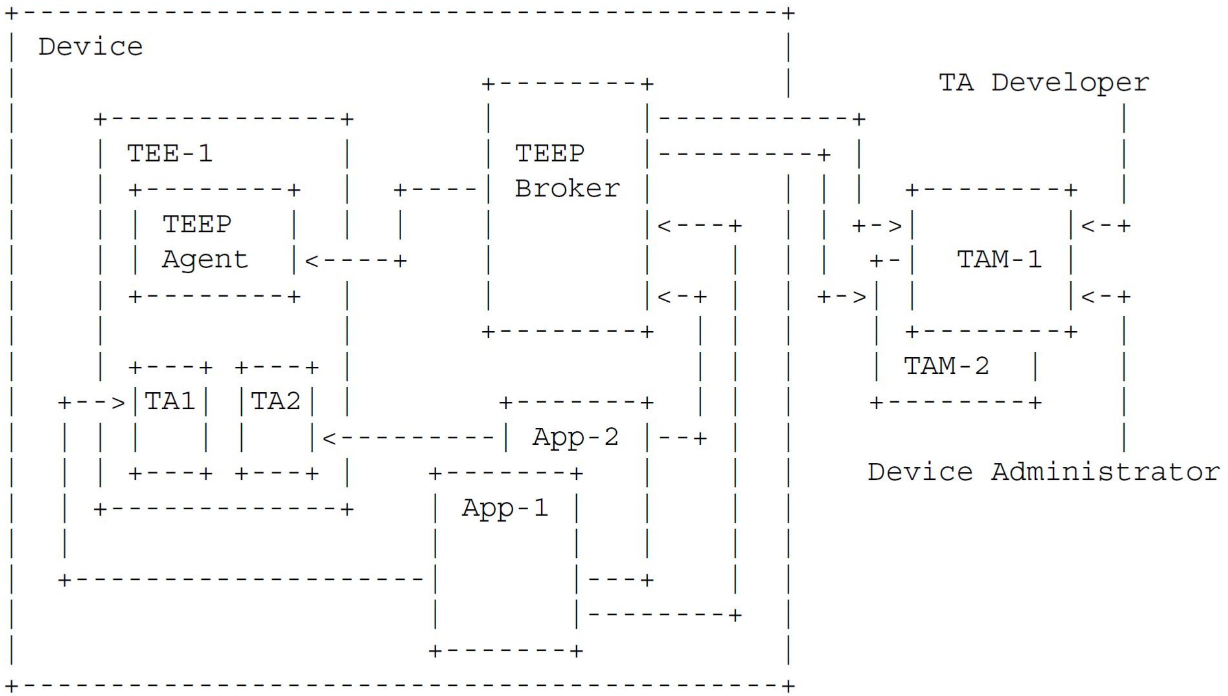
Management of TA (Install/Update/Delete)

- Many vendors would like to install/update/delete Trusted Applications remotelly.
    - Through Internet, with USB stick and etc.
- The mechanism must be secure and trustful. Therefore, the protocol must be defined by the authorized organization.

- IETF has a Working Group for TEEP (Trusted Execution Environment Provisioning)

TEEP from the IETF draft

- Assuring Trusted Application (TA), developed by venders A, to be installed, executed and deleted in secure way on the devices developed by other than vender A. (same vendor is also permitted)
- To achieve the objective of TEEP, utilize TEE hardware mechanism on CPU architecture for executing TAs.

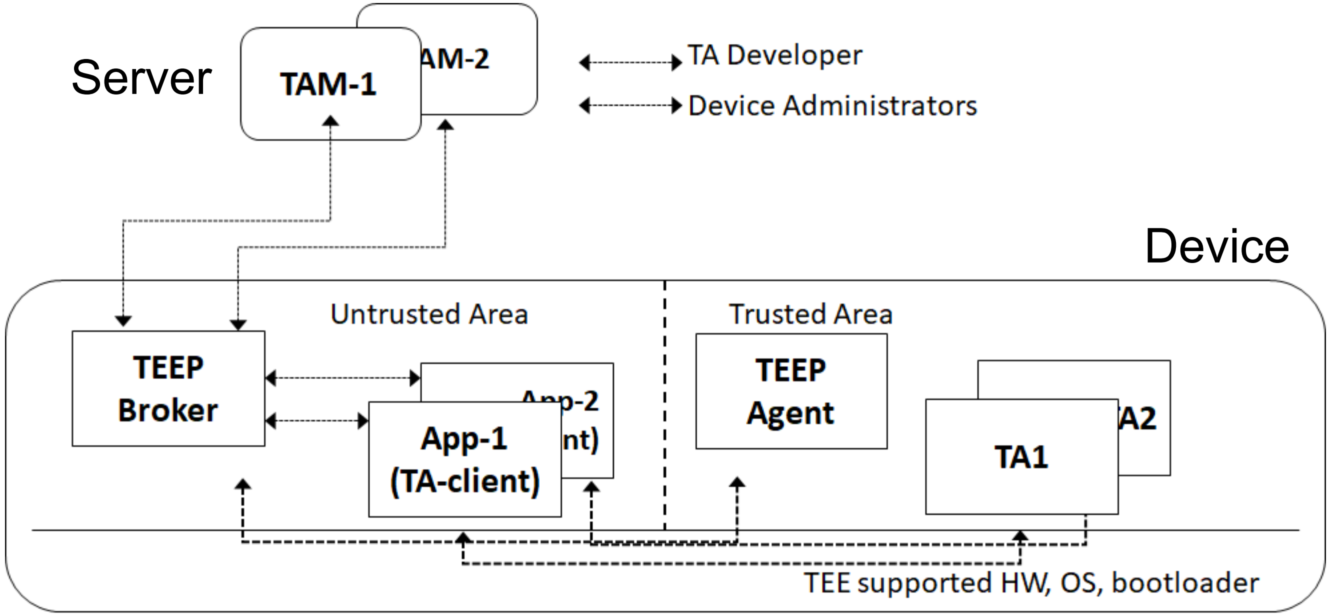


```
+-------------------------------------------+
| Device                                    |
|                          +--------+       |        TA Developer
|    +-------------+       |        |-----------+              |
|    | TEE-1       |       | TEEP   |---------+ |              |
|    | +--------+  |  +----| Broker |       | | |  +--------+  |
|    | | TEEP   |  |  |    |        |<---+  | | +->|        |<-+
|    | | Agent  |<----+    |        |    |  | |  +-|  TAM-1 |
|    | +--------+  |       |        |<-+ |  | +->| |        |<-+
|    |             |       +--------+  | |  |    | +--------+  |
|    | +---+ +---+ |                   | |  |    | TAM-2  |    |
|  +-->|TA1| |TA2| |        +-------+  | |  |    +--------+    |
|  | | |   | |   |<---------| App-2 |--+ |  |                  |
|  | | +---+ +---+ |    +-------+   |    |  |    Device Administrator
|  | +-------------+    | App-1 |   |    |  |
|  |                    |       |   |    |  |
|  +--------------------|       |---+    |  |
|                       |       |--------+  |
|                       +-------+           |
+-------------------------------------------+
```

Figure 1, from TEEP Architecture draft

Simplified TEEP overview

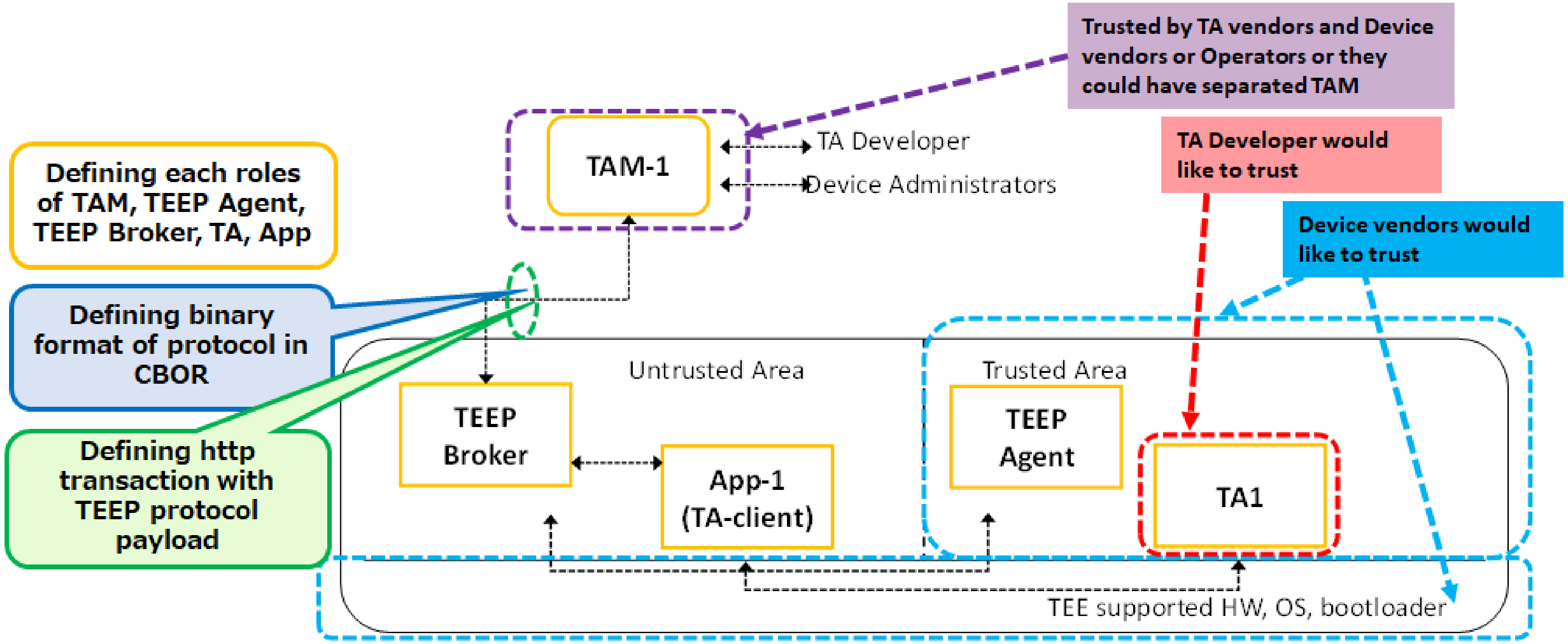
- TAM
  - Manages installing, executing, deleting singed TAs in Devices from remote location.
- TEEP-Agent
  - Verify signed TAs from TAM and handles install, execute, delete TAs inside Device.
  - TEEP-Broker acts proxy between TAM and TEEP-Agent.
- TA and App pairs
  - Handles Secure operations and/or sensitive data
- Trusted Area
  - Only Device vendors and/or TA vendors could install App/Data
- Untrusted Area
  - Users could freely install App/Data. etc Linux, Windows



TEEP coverage among drafts

- Three IETF drafts defining TEEP
  - TEEP Architecture draft
  - TEEP Protocol draft
  - TEEP over http draft

- Prerequisites from other Working Groups
  - SUIT Working Group
    - Defining Manifest format of TA binary
  - RATS Working Group
    - Method of Authenticity of TEE and Device



Defining each roles of TAM, TEEP Agent, TEEP Broker, TA, App

Defining binary format of protocol in CBOR

Defining http transaction with TEEP protocol payload

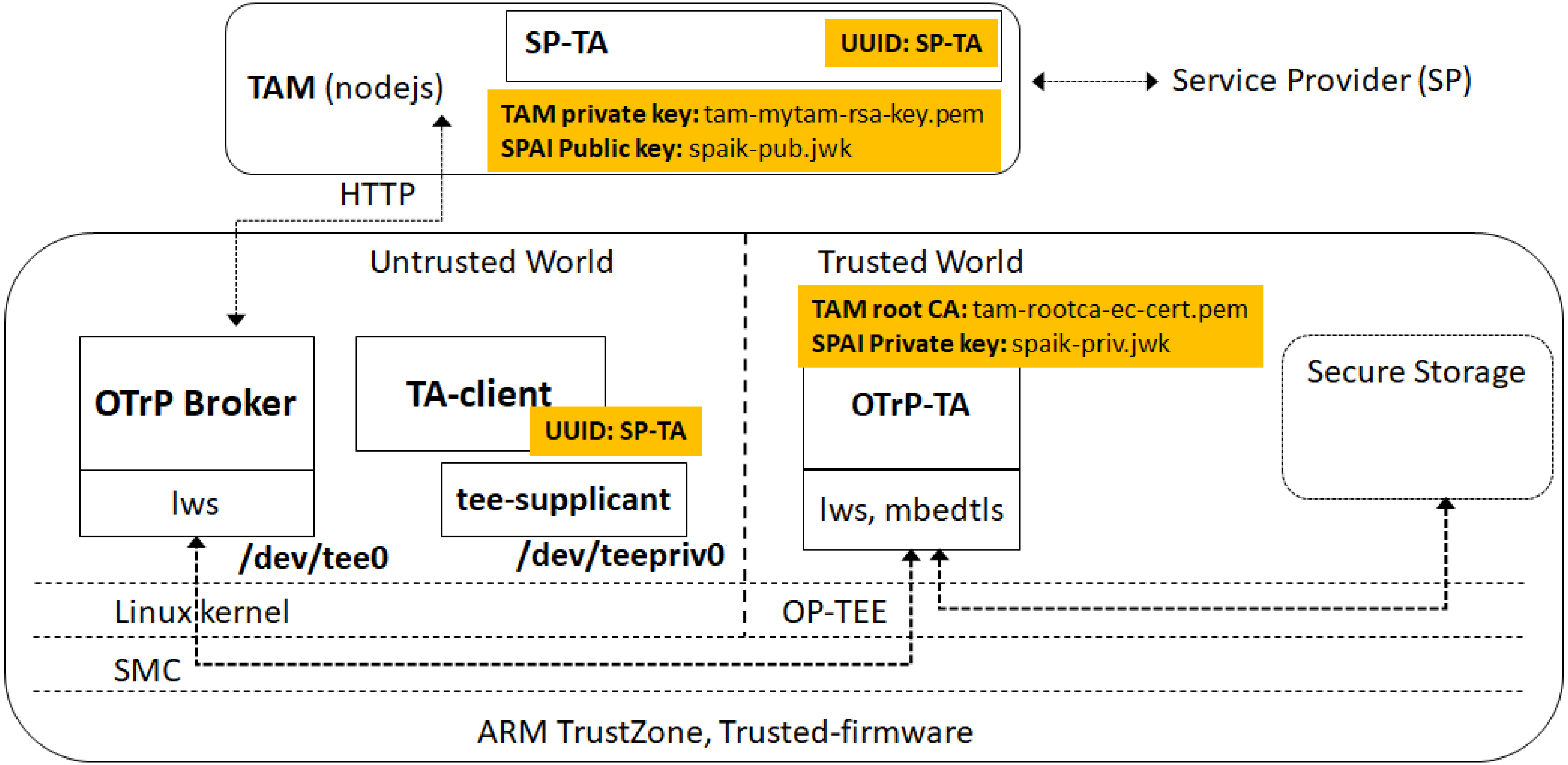
Trusted by TA vendors and Device vendors or Operators or they could have separated TAM

TA Developer would like to trust

Device vendors would like to trust

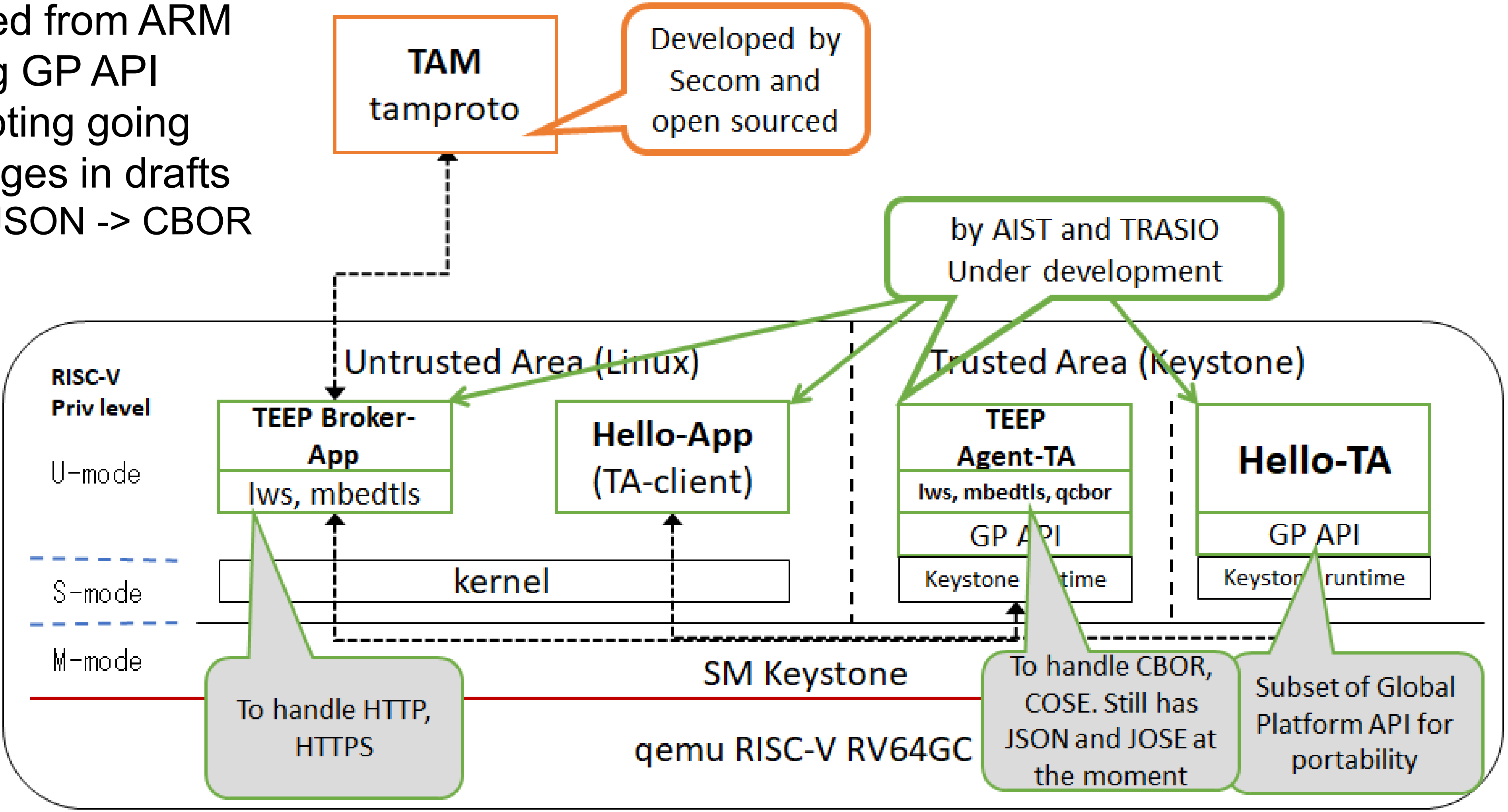
Initial prototype of TEEP on ARM Cortex-A

- Based on old TEEP Architecture draft



Current TEEP implementation on RISC-V

- Ported from ARM using GP API
- Adapting going changes in drafts
  - JSON -> CBOR



TAM
tamproto

Developed by Secom and open sourced

by AIST and TRASIO
Under development

RISC-V
Priv level

U-mode

S-mode

M-mode

Untrusted Area (Linux)

Trusted Area (Keystone)

TEEP Broker-App
lws, mbedtls

Hello-App
(TA-client)

kernel

TEEP
Agent-TA
lws, mbedtls, qcbor
GP API
Keystone runtime

Hello-TA
GP API
Keystone runtime

SM Keystone

qemu RISC-V RV64GC

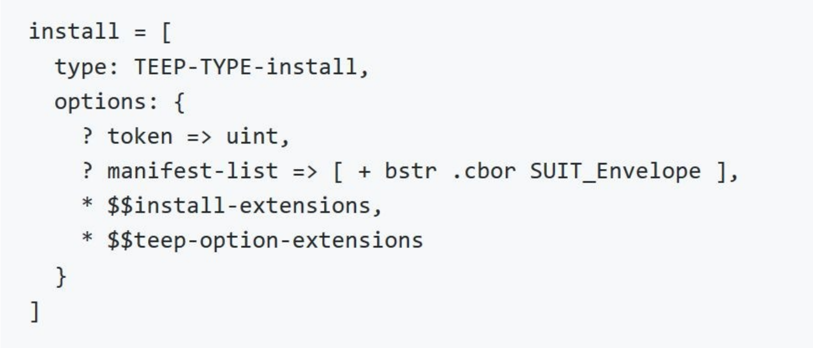
To handle HTTP, HTTPS

To handle CBOR, COSE. Still has JSON and JOSE at the moment

Subset of Global Platform API for portability

Details of TEEP messages

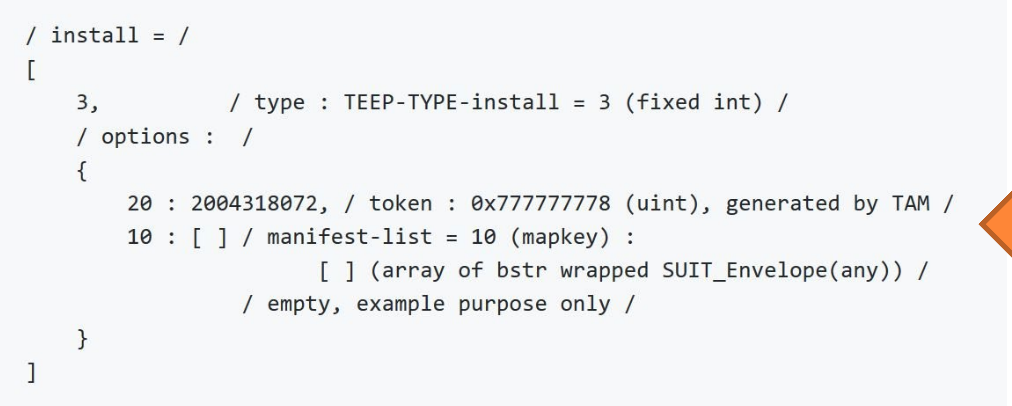
- Concise Data Definition Language (CDDL)



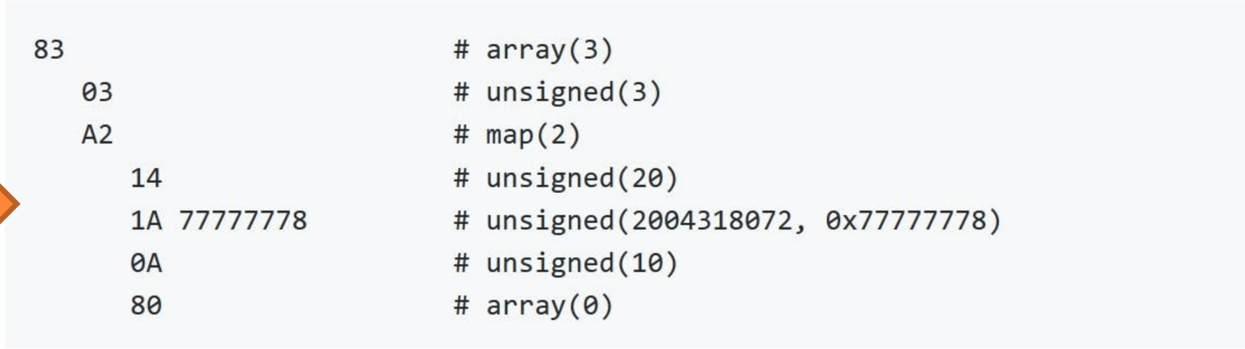
```
install = [
  type: TEEP-TYPE-install,
  options: {
    ? token => uint,
    ? manifest-list => [ + bstr .cbor SUIT_Envelope ],
    * $$install-extensions,
    * $$teep-option-extensions
  }
]
```

- CBOR Diagnostic Notation

CBOR Binary Representation



```
/ install = /
[
    3,          / type : TEEP-TYPE-install = 3 (fixed int) /
    / options :  /
    {
        20 : 2004318072, / token : 0x77777778 (uint), generated by TAM /
        10 : [ ] / manifest-list = 10 (mapkey) :
                       [ ] (array of bstr wrapped SUIT_Envelope(any)) /
                  / empty, example purpose only /
    }
]
```



```
83                 # array(3)
   03              # unsigned(3)
   A2              # map(2)
      14           # unsigned(20)
      1A 77777778  # unsigned(2004318072, 0x77777778)
      0A           # unsigned(10)
      80           # array(0)
```

Summary

- Introduced basic TEE concept
- Importance of TEE for Critical Applications and Operation of Sensitive Data
- Modern CPU Architecture supports TEE
- TEE on RISC-V with Keystone
- IETF is designing and standardizing TEEP for unified way of controlling TAs on different devices and servers
- Relationship of three TEEP drafts
- Status of current development of TEEP on RISC-V
- Having GP API made porting TEEP from ARM to RISC-V easily
- CBOR represnations and binaries

Appendix

- IETF
  - Internet Engineering Task Force
- IETF TEEP Architecture draft
  - https://datatracker.ietf.org/doc/draft-ietf-teep-architecture/
- IETF TEEP Protocol draft
  - https://datatracker.ietf.org/doc/draft-ietf-teep-protocol/
- IETF TEEP over http
  - https://datatracker.ietf.org/doc/draft-ietf-teep-otrp-over-http/
- RATS - Remote ATtestation ProcedureS
  - https://datatracker.ietf.org/wg/rats/documents/
- SUIT - Software Updates for Internet of Things
  - https://datatracker.ietf.org/wg/suit/about/
- CBOR - Concise Binary Object Representation
  - https://datatracker.ietf.org/doc/rfc7049/
- COSE
  - https://tools.ietf.org/html/rfc8152
- RISC-V Keystone project
  - https://keystone-enclave.org/

Updates and discussion at github links

- TEEP Architecture draft
  - https://github.com/ietf-teep/architecture
- TEEP Protocol draft
  - https://github.com/ietf-teep/teep-protocol
- TEEP over http
  - https://github.com/ietf-teep/otrp-over-http

TAM server implementation on github

- https://github.com/ko-isobe/tamproto

Acknowledgement: This presentation is based on results obtained from a project (JPNP16007) commissioned by the New Energy and Industrial Technology Development Organization (NEDO).