IETF 116 TEEP Hackathon

March 27, 2023

Akira Tsukamoto, (presenting)
Dave Thaler, Microsoft
Kohei Isobe, SECOM
Ken Takayama, SECOM
Shin'ichi Miyazawa, SECOM
Yuichi Takita, SECOM
Daisuke Ito, Roboc

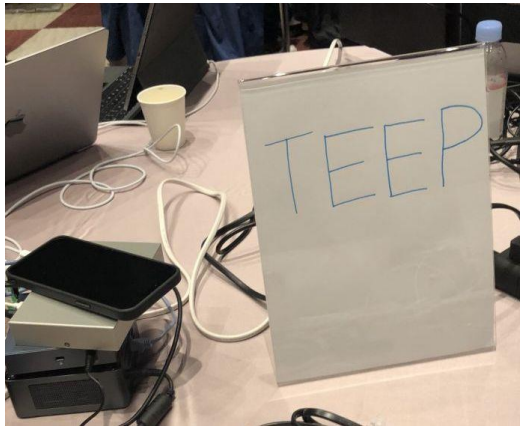
IETF 116 TEEP Hackathon

- Date March 25 Saturday, 26 Sunday
  - Jointly COSE, and TEEP

- Participants:

  Dave Thaler, Microsoft
  Kohei Isobe, SECOM
  Ken Takayama, SECOM
  Shin’ichi Miyazawa, SECOM
  Yuichi Takita, SECOM
  Daisuke Ito, Roboc
  Carsten Bormann, CDDL
  Laurence Lundblade, t-cose
  Akira Tsukamoto

Pictures



Objective and Plan

- Objective
  - Refine the draft from issues found in the implementation

- Action Item list

  - Clarification of cnf recently added to Query Response
    https://github.com/ietf-teep/teep-protocol/pull/321

  - Compromised Broker and keys in multiple TEEP-Agents on SGX
    https://github.com/ietf-teep/teep-protocol/issues/310#issuecomment-1467297393

  - Token and Challenge coexistence in TEEP Messages, from IETF110
    https://github.com/ietf-teep/teep-protocol/issues/127

  - Easy synchronization of cddl definitions between md file and cddl files.
    https://github.com/ietf-teep/teep-protocol/issues/208

- Work on implementations

Clarification of cnf recently added to Query Response
Only for Japan member

- After the consideration of compromised TEEP Agent discussion, the cnf was added to the Query Response.
- PR
  https://github.com/ietf-teep/teep-protocol/pull/321
- The cnf will contains the hash value of public key of the TEEP Agent.
- Among only Japan member, was not sure whether cnf only contain the hash value of TEEP Agent or both TEEP Agent and Verifier.

```
+ The Attestation Result must first be validated as follows:
+
+ 1. Verify that the Attestation Result was signed by a Verifier that the TAM trusts.
+ 2. Verify that the Attestation Result contains a "cnf" claim (as defined in {{Section 3.1 of RFC8747}}) where
+    the key ID is the hash of the TEEP Agent public key used to verify the signature on the TEEP message,
+    and the hash is computed using the Digest Algorithm specified by one of the SUIT profiles
+    supported by the TAM (SHA-256 for the ones mandated in this document).
```

- No, it was misunderstanding. The cnf only contains hash value of TEEP Agent

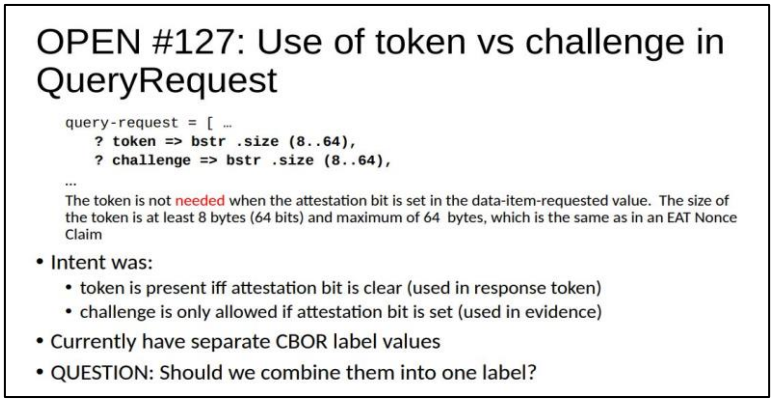
Compromised Broker and keys in multiple TEEP-Agents on SGX

- Initial discussion was compromised Agent.
  https://github.com/ietf-teep/teep-protocol/issues/310
- The TEEP Broker may be compromised but the TEEP Agent itself is protected by SGX.
- When TEEP Broker is compromised, it may have multiple TEEP Agent instances in the same SGX chip.

- Conclusion was we only consider compromised Broker and not Agent in the TEEP design.
- The key pairs are different in different SGX chip which do not contradict with the TEEP design.

Clarification of token and challenge in TEEP Messages

- This topic was resolved once at IETF 110. Revisiting.
- Decision was made to use either of token or challenge at IETF110.



- Always having token may make TAM implementation easier.

- Keep it as it is, and do not change the draft.
- If using timestamp for the freshness, able to reuse AR in QueryResponse.

Synchronizing cddl definitions between md file and cddl files

- Raised between IETF 113 March and IETF 114 July 2022 when attempting cddl syntax check before submitting the draft
  https://github.com/ietf-teep/teep-protocol/issues/208
- The downloading dependent cddl files were fixed between IETF 114 and IETF 115 hackathon.
- The cddl syntax check command in Makefile was added at IETF 115 hackathon.
- When updating md file, it is burden to manually making the same changes to cddl files without making mistakes.

- Updating Makefile to extract cddl file from md file.
- Do not require updating cddl file manually anymore.
  https://github.com/ietf-teep/teep-protocol/pull/322

Benefit of CBOR in TEEP (1/2)

CBOR がバイナリーになるまで (1/2)

- TEEP query-response の例
- CDDL (Concise Data Definition Language)

```
query-response = [
   type: TEEP-TYPE-query-response,
   options: {
    ? token => bstr .size (8..64),
    ? selected-cipher-suite => suite,
    ? selected-version => version,
    ? evidence-format => text,
    ? evidence => bstr,
    ? tc-list => [ + tc-info ],
    ? requested-
    ? unneeded-
    ? ext-list =>
    * $$query-r
    * $$teep-op
   }
 ]
```

- Diagnostic Notation

```
/ query-response = /
{
  2,  / type : TEEP-TYPE-query-response = 2 (uint (0..23)) /
  / options : /
  {
    20 : 0xa0a1a2a3a4a5a6a7a8a9aaabacadaeaf,
         / token = 20 (mapkey) :
           h'a0a1a2a3a4a5a6a7a8a9aaabacadaeaf' (bstr .size (8..64)),
           given from TAM's QueryRequest message /
    5 : 1,  / selected-cipher-suite = 5 (mapkey) :
           TEEP-AES-CCM-16-64-128-HMAC256--256-X25519-EdDSA =
           1 (.within uint .size 4) /
    6 : 0,  / selected-version = 6 (mapkey) :
           0 (.within uint .size 4) /
    7 : ... / evidence = 7 (mapkey) :
           Entity Attestation Token /
```

```
{
  2,  / type : TEEP-TYPE-query-response = 2 (uint (0..23)) /
  / options : /
  {
    20 : 0xa0a1a2a3a4a5a6a7a8a9aaabacadaeaf,
         / token = 20 (mapkey) :
           h'a0a1a2a3a4a5a6a7a8a9aaabacadaeaf' (bstr .size (8..64)),
           given from TAM's QueryRequest message /
```

] / component-id =
0a0b0c0d0e0f']

] / component-id =
0a0b0c0d0e0f']

Benefit of CBOR in TEEP (2/2)

CBOR がバイナリーになるまで (2/2)

- Binary Representation

```
82                  # array(2)
  02                # unsigned(2) uint (0..23)
  A5                # map(5)
    14              # unsigned(20) uint (0..23)
    4F              # bytes(16) (8..64)
      A0A1A2A3A4A5A6A7A8A9AAABACADAEAF
    05              # unsigned(5) uint (0..23)
    01              # unsigned(1) .within uint .size 4
    06              # unsigned(6) uint (0..23)
    00              # unsigned(0) .within uint .size 4
    07              # unsigned(7) uint (0..23)
      ...           # Entity Attestation Token
    08              # unsigned(8) uint (0..23)
    82              # array(2)
      81            # array(1)
        4F          # bytes(16)
          000102030405060708090A0B0C0D0E0F
      81            # array(1)
        4F          # bytes(16)
          100102030405060708090A0B0C0D0E0F
```

- JSON で変換(16進数)　259 Bytes

```
205B0A202020322C0A2020207B0A20202020203230203A203078613061316132613361346135613661376138613961616162616361646165616620C0A20202020203520 3A20312C0A202020202036203A20302C0A20202020203820 3A205B0A20202020202020 7B0A202020202020202020 3136203A205B20307830303031303230333034303530363037303830393061306230633064306530662 05D0A20202020202020 7D2C0A20202020202020 7B0A202020202020202020 3136203A205B20307831303031303230333034303530363037303830393061306230633064306530662 05D0A20202020202020 7D0A20202020202020202020 5D0A20202020207D0A205D0A
```

- CBOR で変換(16進数)　63 Bytes

```
8202A5144FA0A1A2A3A4A5A6A7A8A9AAABACADAEAF050106000882814F00010203040506
0708090A0B0C0D0E0F814F100102030405060708090A0B0C0D0E0F
```

Smaller binary than JSON

Started downloading dependent CDDL files with wget/curl

My procedure of cddl tool usage (1/2)

(1) Install cddl tool

```
$ sudo gem install cddl
```

(2) Prepare other CDDL files required for TEEP Protocol

(a-1) CDDL file for SUIT manifest

```
$ wget https://raw.githubusercontent.com/suit-wg/manifest-spec/master/draft-ietf-suit-manifest.cddl
```

(b-2) Fixing errors temporary by adding four lines to draft-ietf-suit-manifest.cddl just downloaded

```
COSE_Sign_Tagged     = 98
COSE_Sign1_Tagged    = 18
COSE_Mac_Tagged      = 97
COSE_Mac0_Tagged     = 17
```

(c) CDDL file for SUIT_Report
Create suit-report.cddl file by going at https://github.com/ietf-teep/teep-protocol/issues/212

(3) Creating CDDL file of TEEP Protocol

```
$ cat draft-ietf-suit-manifest.cddl suit-report.cddl draft-ietf-teep-protocol.cddl > check-draft-ietf-teep-protocol.cddl
```

(4) Run cddl tool

```
$ cddl check-draft-ietf-teep-protocol.cddl generate
```

16

Added CDDL Syntax check with Carsten's CDDL tool

- Added command 'validate-teep-cddl' in Makefile

To check syntax cddl syntax in TEEP file and not suit which is useful during debugging teep by using only QueryRequest which do not contain SUIT part.

```
make validate-teep-cddl
```

```
.PHONY: validate-teep-cddl
validate-teep-cddl: $(CONCATENATED_CDDL) ../cbor/query_request.diag.bin
	cddl $(CONCATENATED_CDDL) validate ../cbor/query_request.diag.bin
	@echo "Success: QueryRequest message matches TEEP Protocol CDDL"
```

TEEP with Passport model Verifier

Demo (1/3)

ARM OP-TEE

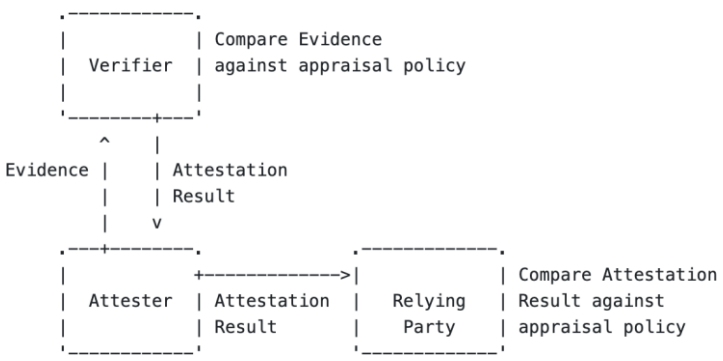
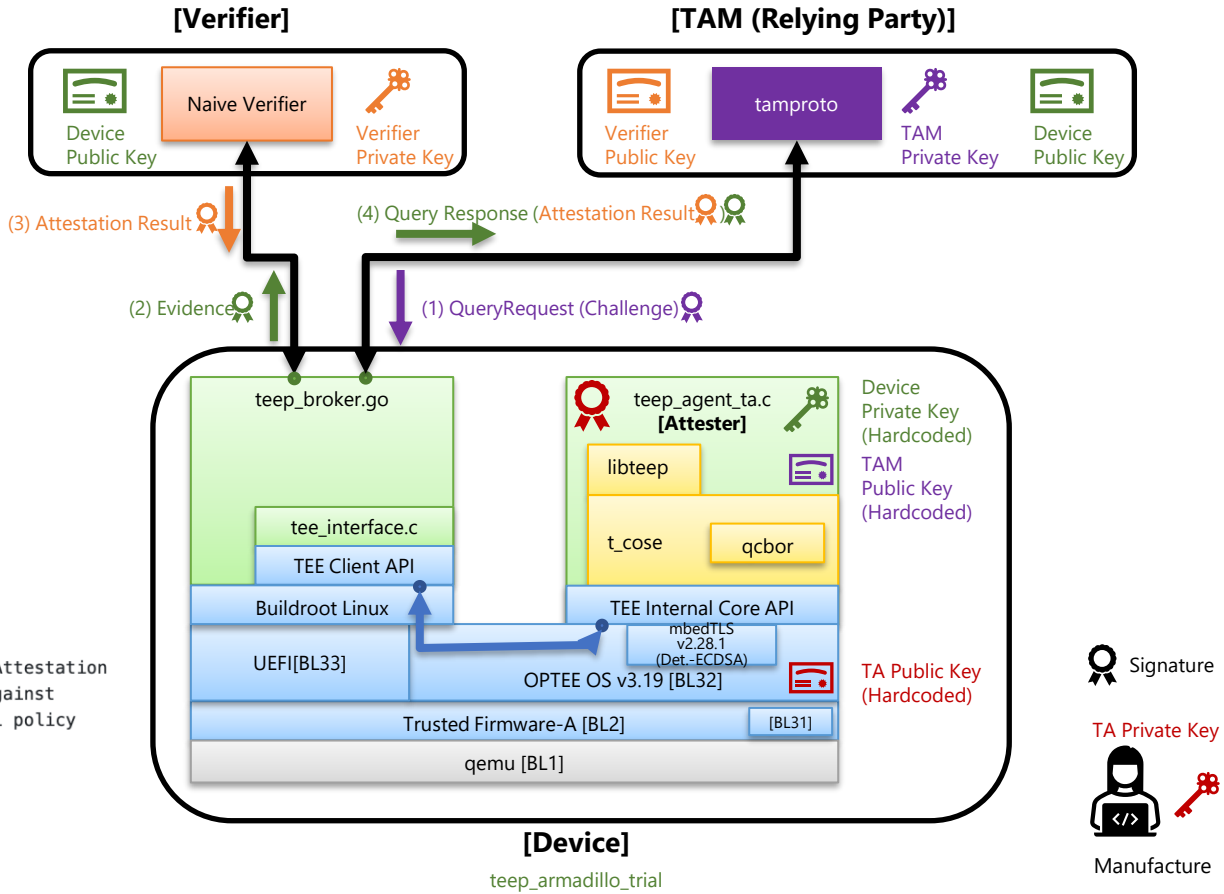


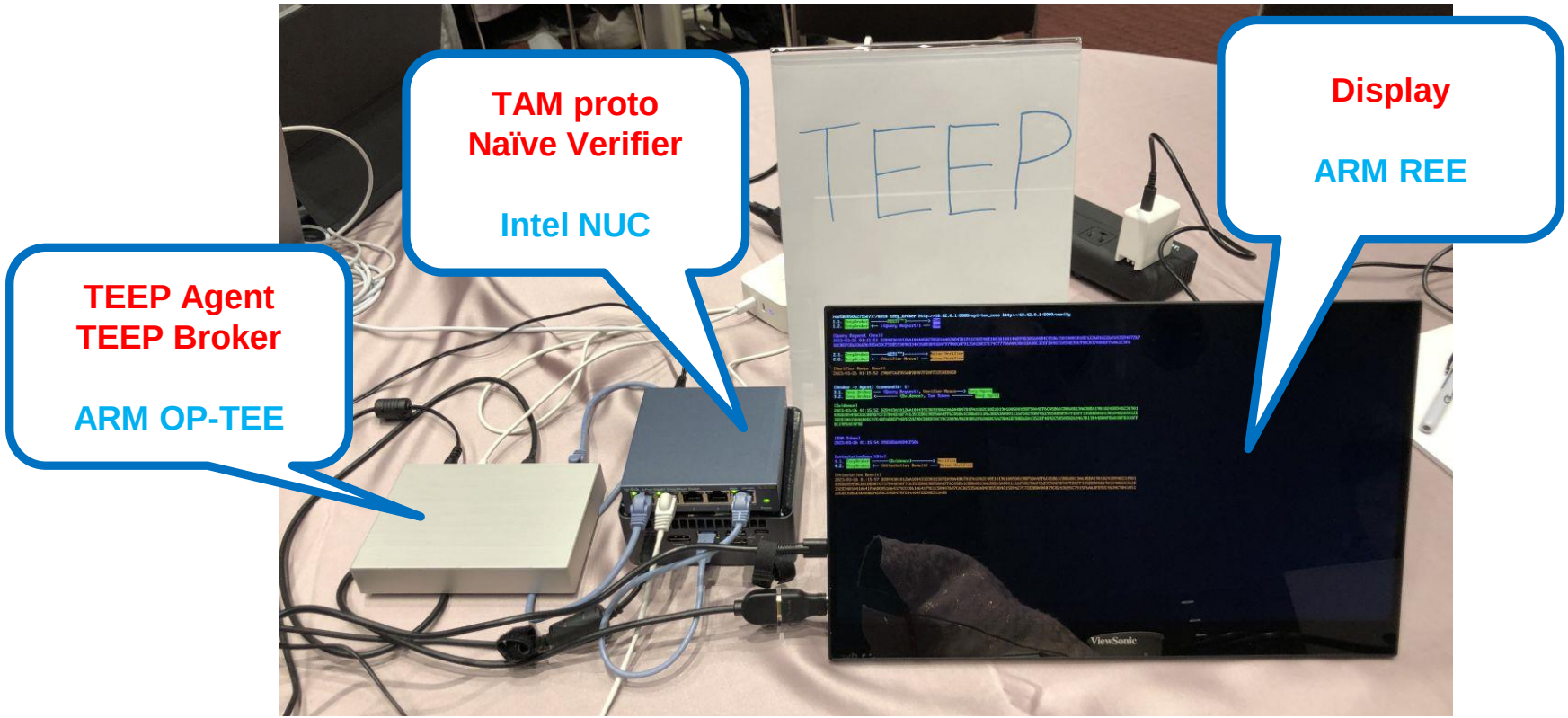
Figure 5: Passport Model

https://datatracker.ietf.org/doc/rfc9334/



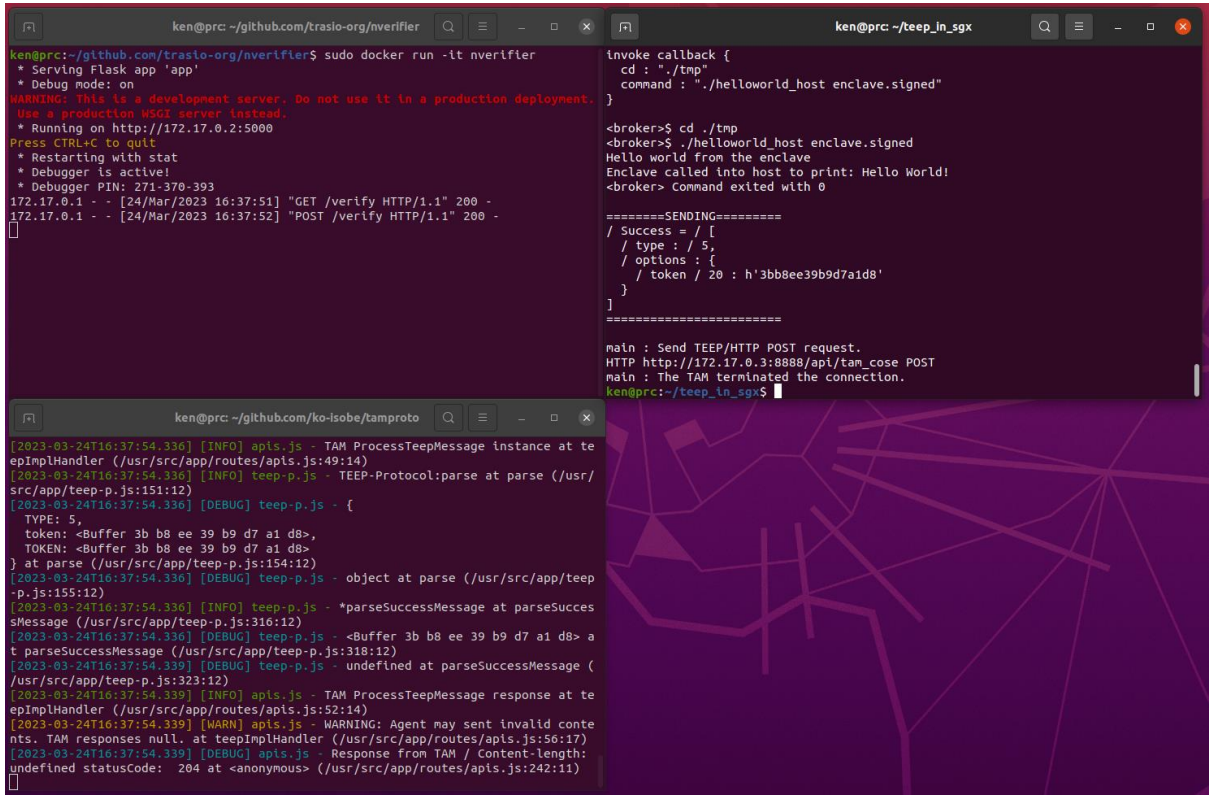
TEEP with Passport model Verifier

Demo (2/3) ARM OP-TEE



TEEP with Passport model Verifier on SGX

Demo (3/3)



Appendix

Items to tackle at Hackathon

- Clarification of cnf recently added to Query Response
  https://github.com/ietf-teep/teep-protocol/pull/321

- Compromised Broker and keys in multiple TEEP-Agents on SGX
  https://github.com/ietf-teep/teep-protocol/issues/310#issuecomment-1467297393

- Token and Challenge coexistence in TEEP Messages, from IETF110
  https://github.com/ietf-teep/teep-protocol/issues/127

- Easy synchronization of cddl definitions between md file and cddl files.
  https://github.com/ietf-teep/teep-protocol/issues/208