TEEP Protocol

draft-ietf-teep-protocol-07

Hannes Tschofenig, Ming Pei, David Wheeler,
Dave Thaler, Akira Tsukamoto

Progress since IETF 111

- Almost all old issues have now been resolved
  - All but ones labeled as “other draft” or “optimization”
  - But some new ones filed
- Most of the spec has now been implemented by at least one implementation

Changes per IETF 111 discussion

- #143: Added error code for no freshness mechanism in common
- #144: Clarified use of challenge when freshness mechanism is not nonce
- #147: Removed use of “TEEP CBOR tag” since always known to be TEEP
- #148: Removed mention of OCSP
  - RECOMMENDED ability to update trust anchor store, e.g., using firmware update
- #150: IANA policy for new ciphersuites and freshness mechanisms
  - Specification Required

- (continued on next slide)

Capability discovery changes per IETF 111

- #145: Removed suit-commands bit in data-item-requested
  - Just use SUIT reports to discover lack of such support when an error occurs
- #146: Clarified handling of TEEP extension negotiation
  - Now consistent with ciphersuite negotiation
- #149: OCSP capability discovery mechanism underspecified
  - Removed since no longer needed

#83: Token Management consideration

- Added elaboration that:
  - Tokens must be random
  - Tokens must be different per request
  - Only one response per token is accepted
  - Tokens must not be used for other purposes
  - Should also have some expiration if not used

#166: When is token included in Update msg?

- Update has optional token, but nothing says when to include it
- Success/Error msgs include token if present in Update
- Suit-reports include suit-report-nonce if present in Update
- Proposal:
  - Explicitly say up to TAM implementation to decide whether to insert a token
    - No constraints
  - TEEP Agent must ensure suit-report-nonce present if Update contains a nonce
    - Already implied but not explicitly stated

Use of EAT

Use of EAT

- Issue #151 (done) and #165 (open):

Requirement from arch draft	RATS claim (draft -07)	Proposal for next draft
Vendor of the device	vendor-identifier [Birkholz]	oemid
Class of the device	class-identifier [Birkholz]	hardware-class (pending EAT PR)
Device unique identifier	device-identifier [Birkholz]	euid
TEE hardware type	chip-version	
TEE hardware version	chip-version	
TEE firmware type	component-identifier [Birkholz]	sw-name (new in EAT-11)
TEE firmware version	version [Birkholz]	sw-version (new in EAT-11)
Freshness proof	nonce	

Legacy OTrP draft said:

- The SP needs to determine security-relevant information of a device before provisioning information to a TEE. Examples include the verification of the device 'root of trust', the **type of firmware** installed, and the **type of TEE** included in a device.

Architecture draft (in IESG) text:

The following information is required for TEEP attestation:

- Device Identifying Information: Attestation information may need to uniquely identify a device to the TAM. Unique device identification allows the TAM to provide services to the device, such as managing installed TAs, and providing subscriptions to services, and locating device-specific keying material to communicate with or authenticate the device. **In some use cases it may be sufficient to identify only the class of the device.** The security and privacy requirements regarding device identification will vary with the type of TA provisioned to the TEE.

Text continued…

- TEE Identifying Information: The **type of TEE** that generated this attestation must be identified. This includes version identification information for hardware, firmware, and software version of the TEE, as applicable by the TEE type. TEE manufacturer information for the TEE is required in order to disambiguate the **same TEE type created by different manufacturers** and address considerations around manufacturer provisioning, keying and support for the TEE.

draft-birkholz-rats-suit-claims has:

```
3.1.2.  class-identifier


   A RFC 4122 UUID representing the class of the
Attester
   or one of its hardware and/or software components.


   $$system-property-claim //= ( class-identifier =>
RFC4122_UUID )
```

- TEEP protocol spec currently references above

EAT PR #139 from Laurence has:

Hardware OEM Class Claim (hardware-class-claim)

This claim value is set by the hardware OEM vendor to indicate the specific hardware model. It is useful for determining software compatibility with the hardware.

There is no global scheme or format for this claim. Each hardware OEM vendor uses it as they see fit.

$$claims-set-claims //= (hardware-class-label => **bytes**)

List discussion on class identifier

- Is it specific to TEEP or general?
- Is the value globally unique, or up to the vendor, or up to the profile?
- Is it opaque or structured?
- Is value a UUID? byte string? text string (as OTrP does)? OID? URL?

#171 EAT Profile Discussion

- draft-ietf-rats-eat section 7 covers requirements for an EAT Profile:
  - Use of JSON, CBOR or both
  - CBOR Map and Array Encoding
  - CBOR String Encoding
  - CBOR Preferred Serialization
  - COSE/JOSE Protection
  - COSE/JOSE Algorithms
  - DEB Support
  - Verification Key Identification
  - Endorsement Identification
  - Freshness
  - Required Claims
  - Prohibited Claims
  - Additional Claims
  - Refined Claim Definition
  - CBOR Tags
  - Manifests and Software Evidence Claims

EAT Profile Discussion

Currently many but not all of those are covered in the TEEP protocol spec.

Should EAT profile be in:

A) TEEP Protocol spec

B) Separate spec

Strawman: (A) same spec

#17 & 40: Unsolicited QueryResponse

- If TEEP Agent already knows TAM key/cert, can you send an unsolicited QueryResponse to avoid the extra round trip?
  - When freshness mechanism is epoch id or timestamp (not nonce)
  - More generally: when QueryResponse contains no information that varies per QueryRequest
- Proposal:
  - TEEP Agent MAY send unsolicited QueryResponse to a TAM if
    - the last QueryRequest contained no token or challenge, AND
    - the TEEP Broker didn't send call ProcessError since the last QueryRequest received, AND
    - TAM key/cert is cached and still valid

Use of SUIT

Use of SUIT manifests

- Brendan's manifest examples were added to draft -07
  - That is sufficient to close issue #41
- However, he also provided example SUIT reports at IETF 111
  - Issue #170 tracks adding these to the TEEP protocol spec

- New issue #158: Three types of SUIT manifests in Update messages
  - (Akira covered)
- New issue #168: Delete a Trusted Component SUIT Manifest…

#168: Removing (unlinking) a component

- Q1: Should a SUIT parser unlink all the dependency manifests when unlinking the dependent manifest?
- Q2: What happens if you try to *locally* run a command to unlink a component? (a la “apt remove –y foo”)

  - Option 1) “Unlink” manifest has largest manifest sequence #
    - This is already the current state, but doesn’t answer Q1
  - Option 2.1) Execute uninstall as reversal of install
    - Won’t work with all SUIT manifests
  - Option 2.2) Add suit-uninstall command to SUIT manifest I-D
    - Original manifest contains (not executed) directives needed to uninstall

Personalization Data scenarios

- A) Developer provides binary and personalization data
- B) Operator/owner provides binary and personalization data
  - Roughly equivalent to case A
- C) Developer provides Trusted Component binary, Operator/owner provides personalization data
  - Personalization data manifest has TC binary manifest as dependency
  - This is the Appendix E.1 example today

Ciphersuites: today in draft-07

Value	Ciphersuite
1	AES-CCM-16-64-128, HMAC 256/256, X25519, EdDSA
2	AES-CCM-16-64-128, HMAC 256/256, P-256, ES256

- A TAM MUST support both ciphersuites.
- A TEEP Agent MUST support at least one of the two but can choose which one.

#167: Simplify Ciphersuites

- Isobe-san filed during hackathon 112:
  - “propose we only use COSE Algorithms Registry instead of creating TEEP original ciphersuites”
  - “the code for the COSE could be exactly reusable on both TEEP messages and the SUIT manifests”

```
;REQUIRED to implement:
   teep-cose-algs /= cose-alg-es256
   teep-cose-algs /= cose-alg-eddsa
;OPTIONAL to implement:
   teep-cose-algs /= cose-alg-ps256
   teep-cose-algs /= cose-alg-ps384
   teep-cose-algs /= cose-alg-ps512
   teep-cose-algs /= cose-alg-rsaes-oaep-sha256
   teep-cose-algs /= cose-alg-rsaes-oaep-sha512
```

SUIT Ciphersuites

- Proposal from SUIT meeting:
  - MUST implement HSS-LMS
    - Quantum resistant, faster verification, but private key requires maintenance
  - SHOULD implement ECDSA
    - Mature tooling
  - MAY implement others: RSA, SHA-512?, SHA3?

TAM override of TC binary URI

- Scenario in issue #104:
  - TC developer creates SUIT manifest with URI of TC binary
  - TAM would like to override the URI, e.g., host the binary itself in a location reachable by the TEEP Agent

- I believe this can be done by TAM creating a new manifest with TAM as signer
  - No document changes needed?
- Issue #105 proposes some way(s) to reference original manifest
  - Binary still signed by original signer, just updated URI signed by TAM

Proposals from issue 105 (Akira, Ken)

```
SUIT_Envelope = {
 suit-authentication-wrapper : {
   [
     / algorithm-id / 2 / "sha256" /,
     / digest-bytes / h'(digest-of-suit-manifest)'
   ]
 },
 suit-manifest : {
   suit-common : {
   },
   suit-install : {
     directive-set-parameters : {
       uri : 'http://example.com/original_tc_signed',
       image-digest : [
         / algorithm-id / 2 / "sha256",
         / digest-bytes / h'(digest-of-image)'
       ]
     }
   }
 },
 // TAM adding Severed suit-install. The device can verify
whether the
TC binary image downloaded from the uri added by TAM  is
an
authentic binary created by TC signer with the image-
digest in the
signed area of the SUIT_Envelope.
 suit-install : {
   suit-directive-set-parameters : {
     uri : 'http://example.com/a_not_tc_signed'
   },
 }
}
```

```
SUIT_Envelope = {
  / manifest / 3:bstr .cbor ({
    / common / 3:bstr .cbor ({
      / components / 2:[
        [h'00']
      ],
      / common-sequence / 4:bstr .cbor ([
        / directive-set-parameters / 19,{
          / uri / 21:Null / Points#3. Clarify the uri
will be overridden below (in severed suit-install). /,
        },
        / directive-override-parameters / 20,{
          / image-digest / 3:bstr .cbor ([
            / algorithm-id / 2 / "sha256" /,
            / digest-bytes / h'(digest of image)'
          ]),
          / image-size / 14:34768
        }
      ])
    }),
    / install / 9:[] / Points#2. The digest of severed
suit-install SHOULD NOT be set in order to change the
uri by TAM's will. /
  }),
  / install / 9:bstr .cbor ([ / Points#1. TAM CAN
change severed and not digested suit-install. /
    / directive-override-parameters / 20,{
      / uri /
21:'http://set.by.not.author.but.tam/firmware',
    }
  }
}
```

Optimizations?

Next steps

- Update draft based on this discussion
- Continue implementation & interop testing
- If # unresolved issues goes to 0, start WGLC around IETF 113?