TEEP Protocol

draft-ietf-teep-protocol-06

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

Changes since IETF 110 / draft-05

- Per resolutions from past IETF TEEP meeting (covered in later slides):
  - #43: Clarified that Success/Error might take some time before being received
  - #129: allow Error in response to QueryRequest
  - #132: Permit future ciphersuites w/o confidentiality
  - #139: Clarify challenge vs token in QueryRequest
  - #131: Added freshness mechanism negotiation

- Will go through the above, then through new issues after that

#43: Success/Error might be delayed

“It is important to note that the TEEP Agent’s Update Procedure requires resolving and installing any dependencies indicated in the manifest, which may take some time, and the resulting Success or Error message is generated only after completing the Update Procedure.

Hence, depending on the freshness mechanism in use, the TAM may need to store data (e.g., a nonce) for some time.”

#129: allow Error in response to QueryRequest

- Section 3 already had text that explicitly allowed Error in response to QueryRequest but sections 6.1 and 6.2 were missing equivalent text.
- Now added, e.g.:

> “When a QueryRequest message is received, the Agent responds with a QueryResponse message **if all fields were understood, or an Error message if any error was encountered**.

#132: Future ciphersuites w/o confidentiality

“Any ciphersuites without confidentiality protection can only be added **if the associated specification includes a discussion of security considerations and applicability**, since manifests may carry sensitive information.

For example, Section 6 of [I-D.ietf-teep-architecture] permits implementations that terminate transport security inside the TEE and if the transport security provides confidentiality then additional encryption might not be needed in the manifest for some use cases.

For most use cases, however, manifest confidentiality will be needed to protect sensitive fields from the TAM as discussed in Section 9.8 of [I-D.ietf-teep-architecture].”

#139: challenge vs token in QueryRequest

IETF 110 got consensus to keep both fields as separate fields

(challenge is for evidence, token is for TEEP message)

- "The token MUST be present if and only if the attestation bit is **clear** in the data-item-requested value."
- "The challenge … MUST be absent if the attestation bit is clear (since the token is used instead in that case)."

- *Note: it's legal to have neither, e.g., if attestation mechanism doesn't require a challenge.*

#131: Freshness mechanism negotiation (1/2)

- Design is based on existing ciphersuite negotiation method
- New proposed IANA registry

Value	Freshness mechanism	Description
1	Nonce	The evidence MUST include a nonce provided in the challenge.
2	Timestamp	A timestamp determined via mechanisms outside the TEEP protocol is used in evidence. Challenge is only needed in the QueryRequest message if a challenge is needed in generating evidence for reasons other than freshness.
3	Epoch ID	An epoch ID determined via mechanisms outside the TEEP protocol is used in evidence. Challenge is only needed in the QueryRequest message if a challenge is needed in generating evidence for reasons other than freshness.

#131: Freshness mechanism negotiation (2/2)

- QueryRequest message added:

```
? supported-freshness-mechanisms => [ + freshness-mechanism ],

"The supported-freshness-mechanisms parameter lists the freshness mechanism(s)
supported by the TAM.

Details about the encoding can be found in {{freshness-mechanisms}}.

If this parameter is absent, it means only the nonce mechanism is supported."
```

- Error message added:

```
? supported-freshness-mechanisms => [ + freshness-mechanism ],

"The supported-freshness-mechanisms parameter lists the freshness mechanism(s)
supported by the TEEP Agent.

Details about the encoding can be found in {{freshness-mechanisms}}.

If this parameter is absent, it means only the nonce mechanism is supported."
```

- We will come back to the last sentence above in just a minute…

Issues found during hackathon

- #143: Need error code for no freshness mechanism in common
- #144: use of challenge when freshness mechanism is not nonce
- #145: handling of suit-commands bit in data-item-requested
- #146: handling of extensions requested is ambiguous
- #149: OCSP capability discovery mechanism underspecified
- #148: Section 4.1.2 missing mention of OCSP stapling data
- #147: What is the "TEEP CBOR tag"?
- #150: IANA registration policy for new ciphersuites and freshness mechanisms

#143: error code for freshness mechanism negotiation

- If fail due to no mechanism in common, what error code do you include?
- Also ambiguity around *absent* supported-freshness-mechanisms vs **supported-freshness-mechanisms:[nonce]**
- Proposed resolution (PR #142):
  - supported-freshness-mechanisms MUST be returned if err-code is ERR_UNSUPPORTED_FRESHNESS_MECHANISMS (a new code)
- Rationale:
  - Consistency with how ciphersuite negotiation works now

#144: use of challenge when freshness mechanism is not nonce

- Section 4.2 wasn’t updated when negotiation was added:
  - “When a challenge is provided in the QueryRequest and an EAT is returned with the QueryResponse message then the challenge contained in this request **MUST be copied into the nonce claim** found in the EAT.”
- Simple editorial oversight
- Proposed resolution:
  - Update above text to match section 8

#145: handling of suit-commands bit in data-item-requested

- Section 4.2 defines a suit-commands bit in the data-item-requested in QueryRequest:
  - “**suit-commands (8)** With this value the TAM queries the TEEP Agent for supported commands offered by the SUIT manifest implementation.”
- However, there's no defined way in the QueryResponse to report this information.
- Options:
  1. Remove the bit and leave it to a future extension if needed
  2. Define a way to report it
- My preference: Option 1. Just use SUIT reports to discover lack of such support when an error occurs.
  - Rationale: least effort for both spec and code

#146: handling of extensions requested is ambiguous

- If a TEEP Agent does not support any extensions, and the extensions bit is set in the QueryRequest, is it
- A) legal for the TEEP Agent to omit ext-list from the QueryResponse, or
- B) required that it MUST be present (and if none supported, would be an empty array)?

- Proposed resolution: B
- Rationale: Consistency with current ciphersuite negotiation

#149: OCSP capability discovery mechanism underspecified

- Section 4.2 says:
  - A TAM can query the TEEP Agent for the support of this functionality via the capability discovery exchange, as described above.
- But nothing in the document about how to report whether one supports OCSP or not
- Also no bit in data-item-requested to query for such a capability
- Options:
  1. Remove the sentence
  2. Specify a negotiation mechanism

#148: Section 4.1.2 missing mention of OCSP stapling data (1/2)

- OCSP stapling data is mentioned in:
  - Section 4.2 (QueryRequest Message), which talks about including it, e.g., "The TAM provides OCSP data so that the TEEP Agent can validate the status of the TAM certificate chain without making its own external OCSP service call."
  - Section 9 (Security Considerations), e.g. "The TEEP Agent SHOULD use OCSP information to verify the validity of the TAM's certificate (as well as the validity of intermediate CA certificates)."
- Section 4.1.2 (Validating a TEEP Message) contains no mention of it.
  - It does say " 5. Follow the steps specified in Section 4 of [RFC8152] ("Signing Objects") for validating a COSE_Sign1 object." but RFC 8152 contains no mention of OCSP.

#148: Section 4.1.2 missing mention of OCSP stapling data (2/2)

- For comparison, the old OTrP spec had text like:
  - “Validate that the request TAM certificate is chained to a trusted CA that the TEE embeds as its trust anchor.
  - Cache the CA OCSP stapling data and certificate revocation check status for other subsequent requests.
  - A TEE can use its own clock time for the OCSP stapling data validation.”
- Do we need anything similar, or a reference to another doc?

#147: what is the “TEEP CBOR tag”?

Section 4.1.1 (Creating a TEEP Message):

- “Prepend the COSE *[COSE_Sign1]* object with the TEEP CBOR tag to indicate that the CBOR-encoded message is indeed a TEEP message.”

Section 4.1.2 (Validating a TEEP Message):

- “Remove the TEEP message CBOR tag and verify that one of the COSE CBOR tags follows it.”

Section 10.4 registers a new CBOR tag whose data item is “TEEP Message”

Issue: use is unclear to implementers

- No cross-reference from 4.1.1 or 4.1.2 to 10.4
- No use in any example or CDDL in doc
- How does it relate to RFC 8152 COSE_Sign1_Tagged?
  - “A tagged COSE_Sign1 structure is identified by the CBOR tag 18. The CDDL fragment that represents this is: **`COSE_Sign1_Tagged = #6.18(COSE_Sign1)`**”

#150: IANA registration policy for new ciphersuites and freshness mechanisms

- The spec proposes two new IANA registries:
  - Section 10.2: Ciphersuite Registry
  - Section 10.3: Freshness Mechanism Registry
- However it does not say what the IANA registration policy is.
- We need to follow RFC 8126 and specify more details for IANA.
  - Any Private or Experimental use permitted?
  - FCFS, Expert Review, Spec Required, RFC Required, IETF Review, or Stds Action?
  - What information must be supplied for future requests?

SUIT Manifest Examples

Next steps

- Take feedback from IETF 111 and update draft
- Continue implementing