TEEP Protocol

draft-ietf-teep-protocol-04
Hannes Tschofenig, Ming Pei, David Wheeler, Dave Thaler, Akira Tsukamoto
Use of tokens (issues 40, 71, 83)

• Current status:
  • All messages from TAM carry a uint “token” that the Agent includes in TEEP response messages
  • Requires TAM to create state to store token, even for QueryRequest
    • QueryRequest is otherwise identical for every client (Agent)
    • QueryResponse includes attestation evidence that already uses separate “challenge” field in QueryRequest to provide replay protection
      • E.g., nonce generated by Verifier (not TAM per se) to appear in EAT
  • Introduces complex question of when to clean up state if Agent doesn’t respond
• Proposal 1: get rid of token in QueryRequest/Response
Is token needed for Install/Delete?

- Success/Error messages now include SUIT Reports with results of install/delete

  ```plaintext
  suit-report = {
    suit-report-manifest-digest => SUIT_Digest,  # identifies what was being (un)installed
    suit-report-manifest-uri => tstr,
    suit-report-records => [ *suit-record ]
  }

  suit-record = {
    suit-record-manifest-id => [ *uint ],
    suit-record-manifest-section => int,
    suit-record-section-offset => uint,
    ( suit-record-component-index => uint //
      suit-record-dependency-index => uint
    ),
    suit-record-failure-reason => SUIT_Parameters,
  }
  ```

- But what about replay protection?
  - Probably applies to SUIT Reports in general
  - Proposal 2: add replay protection to SUIT Report and remove token everywhere
Use of ciphersuites (issues 72, 73)

<table>
<thead>
<tr>
<th>Value</th>
<th>Ciphersuite</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AES-CCM-16-64-128, HMAC 256/256, X25519, EdDSA</td>
</tr>
<tr>
<td>2</td>
<td>AES-CCM-16-64-128, HMAC 256/256, P-256, ES256</td>
</tr>
</tbody>
</table>

...?

? supported-cipher-suites => [ + suite ],

? selected-cipher-suite => suite,

; ciphersuites as bitmaps <- (#72) Why bitmaps when they’re in an array?
suite = $TEEP-suite .within uint .size 8

• (#73) Which ciphersuites are Mandatory To Implement for TAM?
• (#73) Which ciphersuites are Mandatory to Implement for TEEP Agent?
Use of RATS claims (issue 49)

- RATS+TEEP thread: “EAT claims needed by TEEP”
- Unique device identifier: ueid
- Device class: <needed>
  - Laurence “We should come up with a new claim.”
- Hardware version: ean-chip-version (or maybe others)
- Firmware/software identifier & version: <needed for multiple layers>
  - Laurence: “The intent for SW version claims is to use CoSWID.”
    - Seems like submodules would be used to organize claims for layered attestation and all of the above claims can occur in any submodule they need to.
  - Nothing in EAT document about CoSWIDs yet
- Propose: don’t define TEEP-specific claims, use generic if we can
Trusted Component ID’s (issue 70)

• We need a common version-independent ID we can use in RequestTA, UnrequestTA, list of installed TA’s, Delete message, etc.

• SUIT manifest has a version-specific Digest ID, and an optional CoSWID. CoSWID spec says:
  • tag-id (index 0): A 16 byte binary string or textual identifier uniquely referencing a software component. The tag identifier MUST be globally unique. If represented as a 16 byte binary string, the identifier MUST be a valid universally unique identifier as defined by [RFC4122]. There are no strict guidelines on how this identifier is structured, but examples include a 16 byte GUID (e.g. class 4 UUID) [RFC4122], or a text string appended to a DNS domain name to ensure uniqueness across organizations.

• Draft -04 currently says:
  • If Concise Software Identifiers [I-D.ietf-sacm-coswid] are used (e.g., in the suit-coswid field of SUIT manifests), the component-id value is the CoSWID tag-id value.

• Should we make CoSWID mandatory when used in TEEP? Other ideas?
Multi-roundtrip TA installation (issue 43)

• How/where is dependency resolution done?
  • Example: TAM wants to tell Agent to install a TA that depends on Personalization Data from a different TAM

• Options:
  1) TAM resolves dependencies and pushes to Agent in one Install message
     • Problematic for cases where dependency is Personalization Data the TAM can’t see
  2) TAM resolves dependencies and pushes to Agent in separate Install messages
     • Same problem
  3) Agent’s SUIT processor resolves dependencies, and doesn’t respond to initial Install until all dependencies are resolved and SUIT processing completes
     • May take a significant time before Success/Error message generated, does TAM keep state?
  4) Agent resolves dependencies, and sends some progress notification

• Propose option 3
Another case

- Example: untrusted app gets updated such that it replaces dependency on TA1 with a dependency on TA2
  - And space is scarce, there's only space for TA2 if TA1 is removed
- QueryResponse indicates TA1 is unneeded, TA2 is requested
- TAM needs to send Delete(TA1) then Install(TA2)
- Delete's Success response (rather than a QueryResponse) could trigger Install
  - But what if Success is lost or delayed until after a QueryResponse?
- Options:
  1) Provide a way to combine Install/Delete into one message
  2) Only trigger Install or Delete in response to QueryResponse, so need to send another QueryResponse after Install or Delete completes
  3) Provide a way to combine Success/QueryResponse into one message
  4) Allow Install or Delete to be triggered by either a Success or QueryResponse, and try to deal with the extra complexity of using Success to trigger state updates in the TAM