SUIT Hackathon Berlin 2020
Draft Progress – TEEP

• TEEP Component ID structure:
  • [ <TEE ID>, <Authority>, <Security Domain>, <TA name> ]
  • Lazy SD instantiation

• Need TAM URIs

• Need TA example in SUIT draft
Draft Progress – SUIT/RATS integration

• Several EAT claims defined for SUIT
  • Signer ID
    • Component ID
      • Digest
      • Vendor ID (optional)
      • Class ID (optional)
  • Root Manifest URI
    • Absolute URI
      • Template (hex digest appended to template)
  • Root Manifest Digest
Draft Progress – Authentication Wrapper encoding

• The Authentication Wrapper contains COSE objects
  • Wrapping these objects in bstr would probably make them more consumable by COSE libraries
Draft Progress – Digest in COSE payload

- The COSE payload for SUIT is now a SUIT_Digest of the manifest
  - This may help if using EdDSA, since RFC8152 does not support HashEdDSA
  - This helps for modular processing of large PQC signatures
Draft Progress – Reference URI

• It may be helpful to provide a reference URI that can be used to obtain a complete copy of a manifest
• This may be a template so that a hex digest, appended to the URI will resolve to a URI for the complete manifest
Draft Progress – Vendor/Class ID examples

• The examples do not match the CDDL.
  • The Vendor ID & Class ID are parameters in CDDL
  • The examples show vendor ID and class ID conditions taking UUID arguments
Draft Progress – Try Each Examples

- The examples for Try Each do not have bstr wrappers on each sequence, but the CDDL does
Draft Progress – Minimal Loops

• A loop over each component appears to be a good optimization.
  • This satisfies loading a flash component ID into a RAM component ID and having them share a digest.

• Map-Test-Execute has not yet been demonstrated. Not sure if it is needed.
  • There have been requests for prioritized parameter lists, so this may still be needed.