Agenda

• Minor Issues
  • URI vs URI Reference?
  • Future Manifests URI(s)?
  • Integrated Payloads
    • How to digest in-situ
    • Reference format
  • Mandatory-to-implement Signature Algorithm

• Major Issue
  • Document Structure
URI vs URI Reference

• URI Reference is already used in section 7.9.1, so this is mostly an oversight:
  “The URI for a dependency enclosed in this way MUST be expressed as a fragment-only reference, as defined in [RFC3986], Section 4.4.”

• Does general use of URI references create a problem?
  • How does a recipient decide what a reference is relative to? Is this implementation-defined?
Future Manifest URI(s)

• In addition to canonical URI
• List of URIs where a device can find future manifests.
  • Is this component-specific?
  • Should this be overridable?
  • Is this an artifact of secure invocation or of installation?
Integrated Payloads: Digests

• suit-condition-image-digest only refers to component IDs.

• Could we enable computing a digest of an integrated payload?
  • What component id should be used for an integrated payload verification?
  • Should we introduce a new command for this?
    • suit-condition-integrated-image-digest?
Integrated Payloads: References

• Currently, Integrated payload references require a string->int conversion. From draft-ietf-suit-manifest:

    The fragment identifier is the stringified envelope key of the dependency. For example, an envelope that contains a dependency at key 42 would use a URI "#42", key -73 would use a URI "#-73".

• This has disadvantages
  - Possible collisions with future extensions
  - String => integer conversions in manifest processor
Integrated Payload Reference Proposal

• String keys for integrated payloads and dependencies.

• URI fragment-only reference is preferred:
  • Unambiguous
  • Concise

• Possible optimization:
  • Fetch always checks for any URI-Reference in SUIT-Envelope prior to remote fetch
  • Seamless prefetching by distribution infrastructure is possible
Integrated Payload Reference Comparison

v14
• Set Parameter-uri = #24:
  • 21:”#24” = 15 63 233234
  • 5 bytes
• SUIT Envelope key 24:
  • 24 = 18 18
  • 2 bytes

Proposed
• Set Parameter-uri = “#a”:
  • 21:”#a” = 15 62 2361
  • 4 bytes
• SUIT Envelope key “#a”:
  • “#a” = 62 2361
  • 3 bytes
Integrated Payload Reference CDDL

SUIT_Integrated_Payload = (suit-integrated-payload-key => bstr)
SUIT_Integrated_Dependency = (
   suit-integrated-dependency-key => bstr .cbor SUIT_Envelope
)
suit-integrated-payload-key = tstr
suit-integrated-dependency-key = suit-integrated-payload-key
Mandatory-to-Implement Signature Algorithm

• Currently have mandatory-to-implement digest algorithm
• Should we define MTI signature algorithm?
• Which COSE algorithms are MTI?
  • HSS-LMS (-46)?
  • ES256 (-7)?
  • EdDSA (-8)?
• Proposal: make HSS-LMS MTI, others optional.
Documents Structure

• Reviews from outside of WG suggest that manifest specification appears very complex or too complex to implement.

• Proposal:
  • Reduce scope of Manifest Specification, move some topics to extensions
  • Core uses cases:
    • Single image, download and install.
    • Single image, download with swap.
    • Single image, XIP A/B
    • Single image, ram load A/B
    • #2-#4 with two images.
Document Structure Proposal: Core Commands & Parameters

Commands
• condition-vendor-id
• condition-class-id
• condition-image-match
• condition-slot-index
• directive-set-component-index
• directive-try-each
• directive-override-parameters
• directive-fetch
• directive-run
• directive-swap
• directive-copy

Parameters
• parameter-vendor-id
• parameter-class-id
• parameter-image-digest
• parameter-image-size
• parameter-slot-index
• parameter-uri
• parameter-source-component
Document Structure Proposal: Extension Drafts

• Encryption (already in progress)

• Compression and differential update

• Multiple Trust Domains
  • TEEP support
  • Delegation chains
  • Dependencies

• Update Management
  • Additional conditions such as image not-match, use before, minimum battery, update-authorised, version match, component offset, abort
  • Additional directives, such as Wait