draft-ietf-suit-manifest-09

Summary of changes: organization

- Added new metadata structure overview
- Added template for A/B firmware image selection
- Added template names to examples
- Minor changes:
  - Moved encoding descriptions from Commands to Parameters
  - Moved Manifest description under Envelope description
  - Improved cross-referencing
  - Added new definitions of terms
  - Clarified difference between Conditions & Directives

Summary of changes: Encoding 1

- Removed bstr wrappers from:
  - suit-common-dependencies
  - suit-common-components

  After feedback from implementers, these wrappers did not appear to be helping the construction of the parser.
- Removed suit-common-dependency-components

  This element caused a potential implementation pitfall because it was a hint, not authoritative.

Summary of changes: Encoding 2

- Refactored text block

  Text needed to be updated to describe multiple components. Component-specific text elements moved to the component-level map(s)

- Added reporting policy

  Nil arguments to commands replaced with bitfield, that provides hints for attestation. (RATS interwork)

- Added optional report object (*)

  Provides a default encoding for serializing the result of, arguments to, and parameters used by a command

  * Is this the right specification?

Summary of Changes: Examples

- Reduced number of examples
- Goal is to have each template represented in at least one example
- Do we need more examples?
  - Dependency
  - Integrated payload
  - Encrypted payload
  - Encrypted dependency

Next Steps

Question: Packed CBOR

- SUIT Common has similar goals to packed CBOR
- Packed CBOR has only just been adopted by CBOR-wg
- Packed CBOR missing features needed by SUIT
  - E.g. Nested Packed CBOR objects
  - E.g. Packing CBOR sequences
  - E.g. Pull-parser-friendly encoding
- We could:
  - Make no change & apply packed CBOR when available
  - Make no change now, begin work on SUIT v2
  - Adopt packed CBOR & simplify manifest (delays SUIT)

Question: COSE Authentication

- COSE signature and MAC objects contain a SUIT_Digest
  - Enables modular processing of large signatures
- Should this be detached?
  - Saves space in multiply-authenticated manifests

Question: Vendor ID

- Currently set to uuid5(NAMESPACE_DNS, “vendor.domain”)
  - Advantages: Free (with domain registration), fixed-size, unique, computable
  - Disadvantages: no lookup, domain changes break computation, 16 bytes
- Proposed: Private Enterprise Number
  - Advantages: Free, small, reverse lookup
  - Disadvantages:
    - No defined UUID Name Space Identifier computation scheme
    - Not just a number, need sub-registrations. E.g. conglomerates
    - Still not robust across spinouts
- Remember: it’s not intended to be human readable; that’s what text is for.

Question: Encoding choices 1

- SUIT broadly reserves negative numbers for customization without standards action or IANA registration.
  - Is this the right choice?
- We lack a way to specify some, but not all components/ dependencies. Propose:
  - set-component-index => [+ uint]
  - set-dependency-index => [+ uint]
  - Only relevant for >= 3 components/dependencies

Question: Encoding choices 2

- Minor Questions:
  - Is there value in separating Conditions from Directives (positive vs. negative) instead of retaining negative numbers for custom Commands/Parameters? Authors have no strong opinion.

    Makes it easy to distinguish between Conditions (no side-effects) and directives (side effects)
  - Should Abort be a condition?

    Behaves just like a failed condition, so should it be classed as one? This is a label change only.

Question: Example formatting (bstr .cbor)

Current Formatting

```
/ payload / h'820258404b4c7c8c0fda76c9c9591a9db160918e
2b3c96a58b0a5e4984fd4e8f9359a928' / [
  / algorithm-id / 2 / "sha256" /,
  / digest-bytes /
h'4b4c7c8c0fda76c9c9591a9db160918e2b3c96a58b0a5e4984fd
4e8f9359a928'
] /,
```

Proposed

```
/ payload / bstr .cbor ([
    / algorithm-id / 2 / "sha256" /,
    / digest-bytes /
h'4b4c7c8c0fda76c9c9591a9db160918e2b3c96a58b0a5e4984fd
4e8f9359a928’
]),
```

Questions: TEEP

- Do we have enough to support TEEP?
  - TA id’s => SUIT component id’s
  - TAM URIs => SUIT uri/uri list parameters
  - Personalization Data => Dependent manifest with TA dependency

Questions: RATS

- Do we have enough to support RATS?

  To facilitate construction of Reports that describe the success, or failure of a given Procedure, each command is given a Reporting Policy. This is an integer bitfield that follows the command and indicates what the Recipient should do with the Record of executing the command. The options are summarized in the table below.

Policy	Description
suit-send-record-on-success	Record when the command succeeds
suit-send-record-on-failure	Record when the command fails
suit-send-sysinfo-success	Add system information when the command succeeds
suit-send-sysinfo-failure	Add system information when the command fails