CoRIM: Modeling, Evolution, and Revocation

Dhawal Kumar | 21-Oct-2025

Agenda

👉 Modeling Scenarios & Verifier Lookup (MSVL)

- Scenario 1: Mutually Exclusive Device Modes
- Scenario 2: Role-Based Claims (User vs. Owner)
- Scenario 3: Pre-Boot (Partial) States
- Challenge: Historical / Dynamic Events
- Recommendation: Add Examples to RFC

👉 RFC Evolution & Tooling (RET)

- Backward Compatibility
- Tooling & Migration Strategy

👉 CoRIM Revocation

👉 Open Discussion & Next Steps

[MSVL] Scenario 1: Non-Coexistent Device Modes

Modeling

- **Scenario**
  - A device has 4 mutually exclusive modes (e.g., CC, non-CC-optimized-perf, non-CC-max, pre-boot).
  - The set of possible measurements (i.e., the keys or types of measurements) is the same for all modes, but the reference values will differ.

- **Question**: How is this scenario best modeled to differentiate these modes?
  1. corim-map.profile: A single CoRIM for each mode (profile corresponds to the mode)?
  2. concise-mid-tag.tag-identity.tag-id: A separate CoMID tag for each mode, all bundled in one CoRIM?
  3. concise-mid-tag.triples.reference-triples.ref-env.class.class-id: A single CoMID, with a different class-id in the environment-map for each mode?

- **Discussion**
  1. Do all options work?
     - Option (1) seems too high-level (at the CoRIM wrapper). But may be right if the “mode” applies to SW (CoSWID). What is the intended use of profile?
     - Option (2) keeps management of “mode” simple by using a different CoMID tag per “mode” instead of repeating the description of “mode” with every entry in triples-map (reference-triples, endorsed-triples, coswid-triples, etc.)
     - Option (3) is most granular but granularity may be overkill
  2. Are there specific scenarios in which each of the 3 options above are best?

[MSVL] Scenario 1: Non-Coexistent Device Modes

Verifier Lookup

- **Question**: Given the modeling choice, how does the Verifier find the right reference values?

- **Dhawal's assumed Verifier Process**
  1. Find CoRIM(s): Look up based on the following from attester cert chain:
     a. Device Make, Device Model, optionally Device Serial. This can be obtained from SPDM/DMTF extension in leaf cert.
     b. Attester ID (e.g., combination of all relevant FWIDs from DiceTcbInfo).
  2. The lookup may yield multiple CoRIMs, CoMIDs, or reference-triples.
  3. Match Evidence: Verifier matches received evidence against the filtered set.

- **Discussion**
  1. Is the Verifier Process as expected? If not, what is expected to be different?
  2. This process must
     a. unambiguously resolve to a single set of reference values (a single "mode").
     b. Yield a unique "mode" (e.g., "CC" or "non-CC-perf") that verifier can return to the Relying Party.
  3. Does "mode" need to be present in the evidence?

[MSVL] Scenario 2: Different Claims for User vs. Owner

Modeling & Verifier Lookup

- **Scenario**
  - A device in a single mode reports different claim sets based on the requestor's role (e.g., device user vs. device owner).

- **Rationale**: Privacy, abstraction of low-level details.
  - A device owner may want privacy around its policies or intend to abstract low level details of policies used to configure the device which device user has no use for.
  - A device user may use some code whose measurement has no use for device owner.

- **Question**:
  1. Is this modeled the same way as Example 1? E.g., Using different class-ids for "user-view" and "owner-view"?
  2. Does "view" or "role" need to be present in the evidence?

[MSVL] Scenario 3: Pre-Boot (Partial) State

Modeling & Verifier Lookup

- **Scenario**
  - Pre-boot state has an incomplete set of measurements. More measurements become available post-boot (e.g., after driver load).

- **Question:** Should this case be viewed as subset of Scenario 1 (i.e. just another distinct "mode")?
  - The reference triple for "pre-boot" would simply contain fewer measurements.

[MSVL] Challenge: Historical/Dynamic Events

Modeling & Verifier Lookup

- **Scenario**
  - A driver is loaded/unloaded multiple times.
  - RTS "extend" functionality means the final PCR/measurement value depends on this history.
  - Ref-Value = f(driver_hash, num_loads)
  - This is further complicated if multiple drivers are loaded/unloaded in different sequences.

- **Problems**
  - A static reference-triple in CoRIM cannot model this dynamic value.
  - The Verifier needs the history of events (the "event log") from the Attester to recalculate the expected measurement.

- **Questions**
  1. Is this "event log" attestation model in scope of CoRIM?
  2. If yes, how are historical/dynamic reference values modeled?
  3. Is the expectation that CoRIM only provides the "golden" hash for the driver binary, and the Verifier's policy dictates how to handle the event log replay?

[MSVL] Recommendation: Add Examples to RFC

Ensuring Consistent Interpretation of the Spec

- **Proposal**
  - Incorporate scenarios like the ones from previous slides (modes, roles, pre-boot) as explicit, normative examples within the RFC text.

- **Rationale**
  - Ensures consistent interpretation and implementation.
  - Reduces ambiguity for implementers (Verifiers, Attesters, CoRIM Generators).
  - While tooling is helpful, the RFC is the ground truth for interoperability.

[RET] Backward Compatibility

Stability & ongoing cost

- **Context**
  - CoRIMs (based on Draft 08 and older) have already been adopted due to the desire for standardization.
  - Backwards incompatible changes can lead to
    - costly-to-fix changes as adoption grows
    - need to maintain multiple incompatible versions of CoRIMs and tooling

- **Questions for Authors**
  1. What is the expected nature of changes before Draft 08 becomes a standard? (e.g., additive, major structural changes, bug fixes?)
  2. Can we "front-load" any remaining backward-compatibility-breaking changes (e.g., before EOY)?

[RET] Tooling & Migration Strategy

Stability

- **Context**
  - Early adoption relies heavily on tooling.

- **Discussion Points**
  1. Is there an official stance on "upgrade" paths for existing CBOR CoRIMs as new drafts emerge?
  2. Can tooling support transformation? (e.g., CBOR (draft 08) -> JSON (tool) -> CBOR (draft 09))
  3. **Suggestion**: Can tooling maintain stability in its own intermediate formats (like JSON templates)?
     - Benefit: This would allow users to re-use their old JSON templates with the new tool to re-generate an updated CBOR, which is much easier than migrating the JSON templates themselves.

CoRIM Revocation

Handling Superseded or Incorrect Manifests

- **Scenarios**
  1. **Superseded**: A new CoRIM is issued (e.g. new CoRIM has more reference values).
  2. **Mistake**: A wrong CoRIM was signed and published.
     1. Business needs may force usage of production PKI to sign pre-release candidates that are found to be incorrect after verification

- **Observation**
  - corim-map.rim-validity provides an expiry date, but no active revocation mechanism before that date.
  - concise-mid-tag.linked-tags (tag-rel: replaces) seems to handle revocation for scenario 1, but not for the entire CoRIM wrapper.

- **Question**
  1. Is there a planned mechanism for CoRIM revocation similar to CRL/OCSP?
  2. Is “tag-rel: replaces” intended to be **the** revocation mechanism? If so,
     1. How is scenario 2 expected to be handled – deletion of the CoRIM?
     2. If deletion of CoRIM is the method, how does publisher ensure that the news has reached all recipients?
     3. Is the expectation that Verifier will get multiple matches when looking up CoRIM and expected to discard a CoRIM that has been replaced?

Open Discussion

Key Questions

1. What is the intended best practice for modeling device "modes"?
2. How should a Verifier link evidence to a "mode" and report that context to the RP?
3. How does CoRIM handle "historical event" measurements (e.g., event logs)?
4. What is the plan for backward-breaking changes before standardization?
5. What is the recommended strategy for CoRIM revocation?


NVIDIA