CoSERV: Concise Selector for Endorsements and Reference Values

Problem Statement

- Conveyance of Endorsements and RVs to the Verifier is within the scope of RFC9334
- It is also a proven industry requirement, evidenced by services such as NVIDIA RIM and AMD KDS
- Existing solutions exhibit fragmentation - proprietary APIs and disjoint data formats
- Fragmentation worsens as such services become more numerous - makes life hard for Verifiers
- How does RATS offer better industry guidance?

Why CoSERV?

- We could define more reference interaction models, or even include concrete API definitions (OpenAPI specs).
- Any such API would need a data format to transact - a query/response language tailored towards RATS artefacts
- This is a good, self-contained thing to define in a draft and validate through prototyping
- Enter CoSERV - the Concise Selector for Endorsements and Reference Values

Requirements and Guiding Principles

- Decoupling of message and transport (*e.g.* could transact over HTTP/REST or CoAP)
- Adaptable to different interaction models (*e.g.* fetch everything, fetch deltas, pub/sub)
- Efficient use of the transport (*e.g.* amenable to client-side or server-side caching)
- Support constrained consumers - optimise data volume and minimise client-side processing burden

Requirements and Guiding Principles (*cont.*)

- Specialise for "RATS-native" artefacts (endorsed values, reference values, trust anchors), but allow some extensibility
- Re-use existing RATS designs where possible, and align to CDDL data model
- Allow distributors to aggregate multiple sources, with flexible trust models

In a nutshell

Self-contained CBOR data item to model an "endorsement" query and the optional result set

```
coserv = {
  &(profile: 0) => profile
  &(query: 1) => query
  ? &(results: 2) => results
}
```

Basic building block for an "endorsement distribution API"

CoSERV Queries

```
query = {
  &(artifact-type: 0) => artifact-type
  &(environment-selector: 1) => environment-selector-map
}
```

What artifact you are interested in (RVs, TAs, etc.)?

The attesters you are interested in?

Query Selectors

```
environment-selector-map = { selector }

selector //= ( &(class: 0) => [ + class-map ] )
selector //= ( &(instance: 1) => [ + $instance-id-type-choice ] )
selector //= ( &(group: 2) => [ + $group-id-type-choice ] )
```

Query Selectors Semantics

Class selectors:

```
SELECT *
  FROM reference_values
 WHERE ( class-id = "iZl4ZVY=" AND class-vendor = "ACME Inc." ) OR
       ( class-id = "31fb5abf-023e-4992-aa4e-95f9c1503bfa" )
```

Instance selectors:

```
SELECT *
  FROM endorsed_values
 WHERE ( instance-id = "At6tvu/erQ==" ) OR
       ( instance-id = "iZl4ZVY=" )`
```

Result Set

```
results = {
  result-set
  &(expiry: 10) => tdate ; RFC3339 date
}

result-set //= reference-values
result-set //= endorsed-values
result-set //= trust-anchors
```

Expressed as CoRIM triples + authority (quads)

Reference Values' Result Set

For example:

```
refval-quad = {
  &(authority: 1) => $crypto-key-type-choice
  &(rv-triple: 2) => reference-triple-record
}

reference-values = (
  &(rvq: 0) => [ * refval-quad ]
)
```

Prototyping

- Veraison has a rich history as a proving ground for RATS designs
- Was built as a Verifier, but can be re-purposed as an Endorser or Reference Value Provider with new API endpoints, transacting CoSERV
- PoC exercise is currently in flight
- More details in a follow-up presentation

Pointers

- corim git:(coserv)
- services git:(coserv)
- I-D.howard-rats-coserv