Project Veraison

[pronounced “verr-ayy-sjon”]

Attestation Verification Components

About Me

Yogesh Deshpande
Principal Engineer
Arm

Security Architect in Architecture and Technology Group

Yogesh.Deshpande@arm.com

https://www.linkedin.com/in/yogesh-deshpande-1454b71/

Main Areas

- Attestation Standards in IETF and TCG
- Core contributor to Project "Veraison"
- Involved in Supply Chain Security

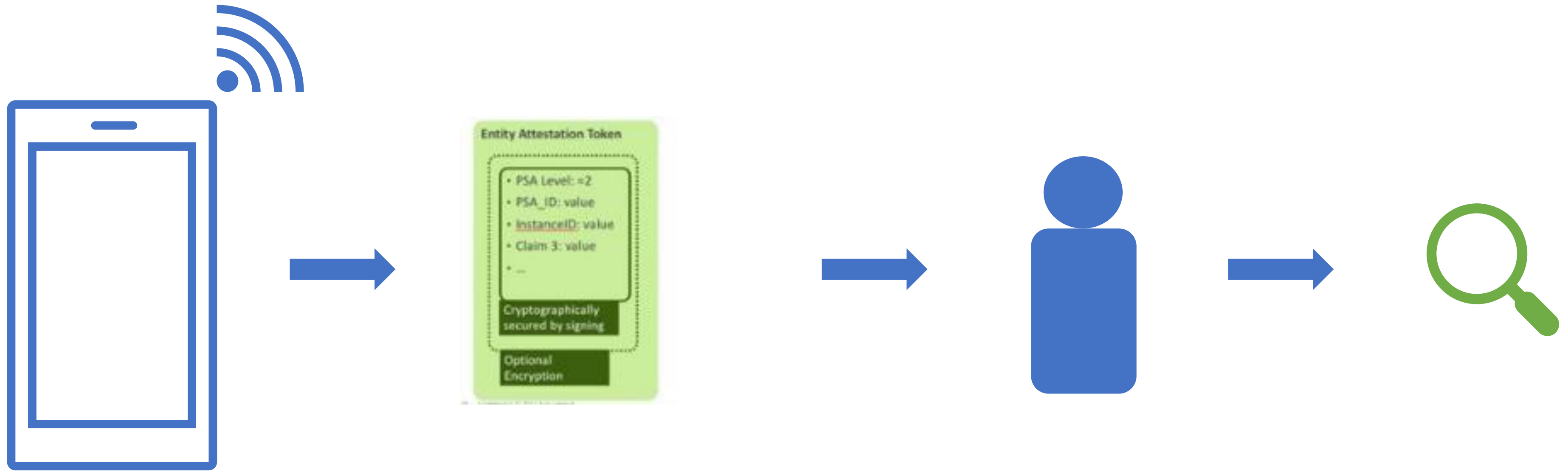
VERAISON

Agenda

- Introduction
- Need for Veraison
- Veraison Architecture
- Libraries and tooling provided by the Veraison Project

What is Attestation?

- A means to establishing the trustworthiness of an entity
- Produces a signed evidence about an entity
- Attestation report alone is insufficient
  - ➢ Must be verified by a trusted service
  - ➢ Verification is at the centre of any attestation flow



VERAISON

The Need for Veraison

VERAISON

Building an Attestation Verification Service

Challenges:

- Due to specific needs of deployments, it is difficult for a single offering to serve all use cases
  - required business relationships
  - regulation / compliance / geo-specifics
- If Verifiers have to be custom, then
  - standardisation and quality levels suffer between deployments
  - the cost of building a trustworthy infrastructure becomes a notable barrier to entry

VERAISON

Building an Attestation Verification Service

Solution:

Make common components available which make building Verification Services more straightforward!

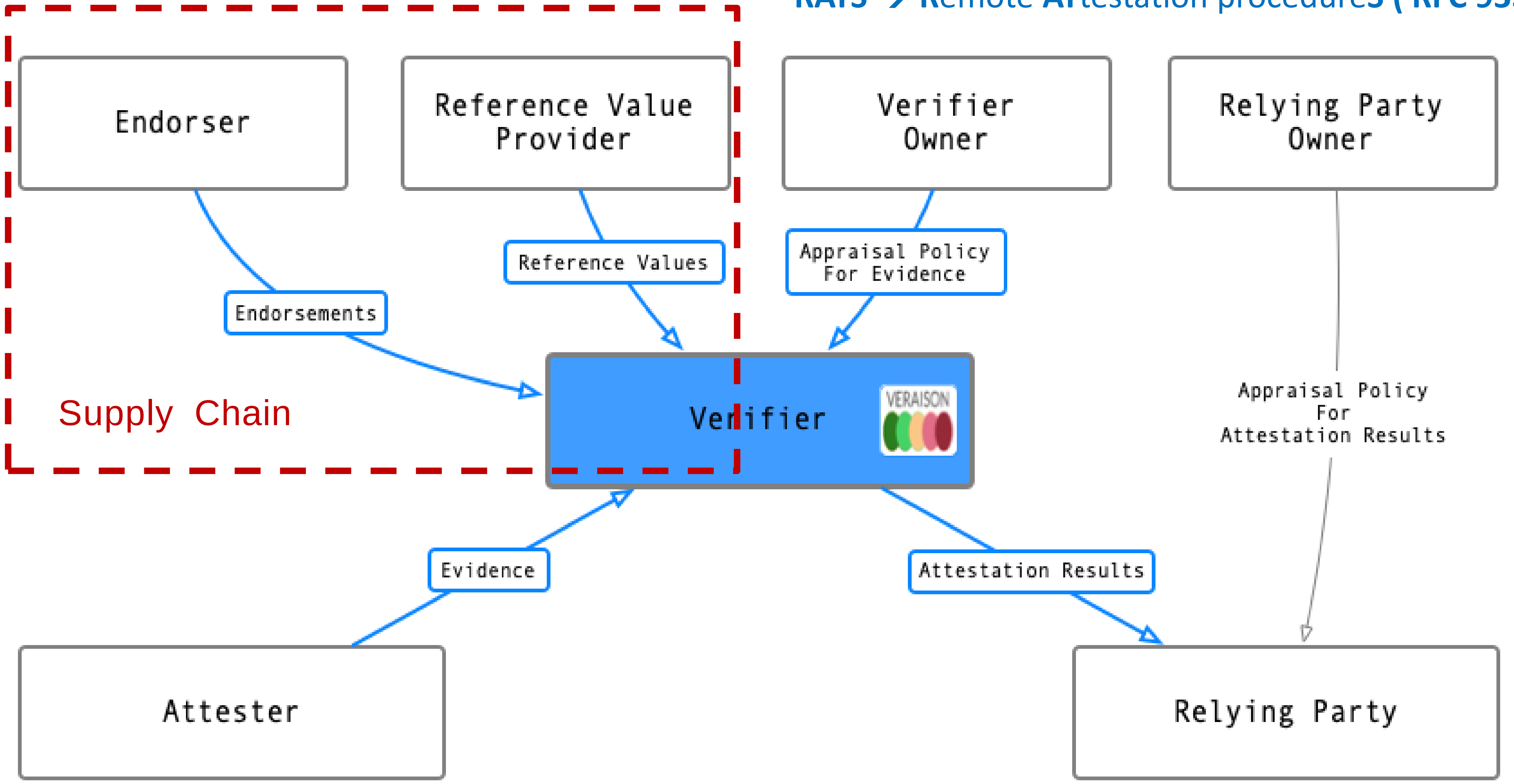
VERAISON

VERific**A**t**I**on of atte**S**tati**ON**

https://github.com/veraison/

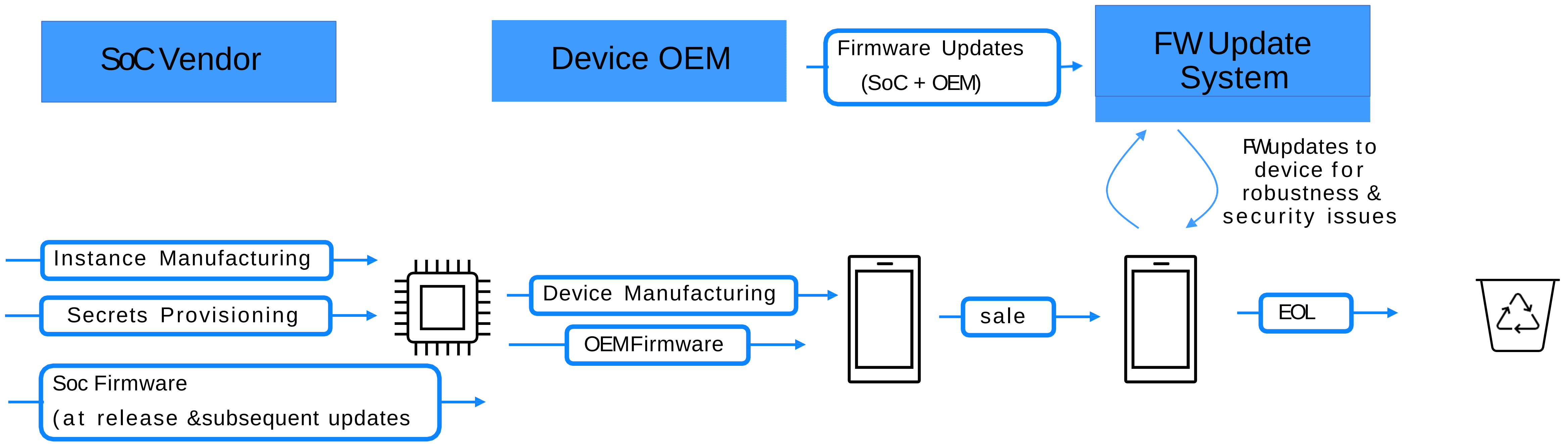
RATS Architecture Model

RATS → Remote ATtestation procedureS (RFC 9334)



Supply Chain & Lifecycle

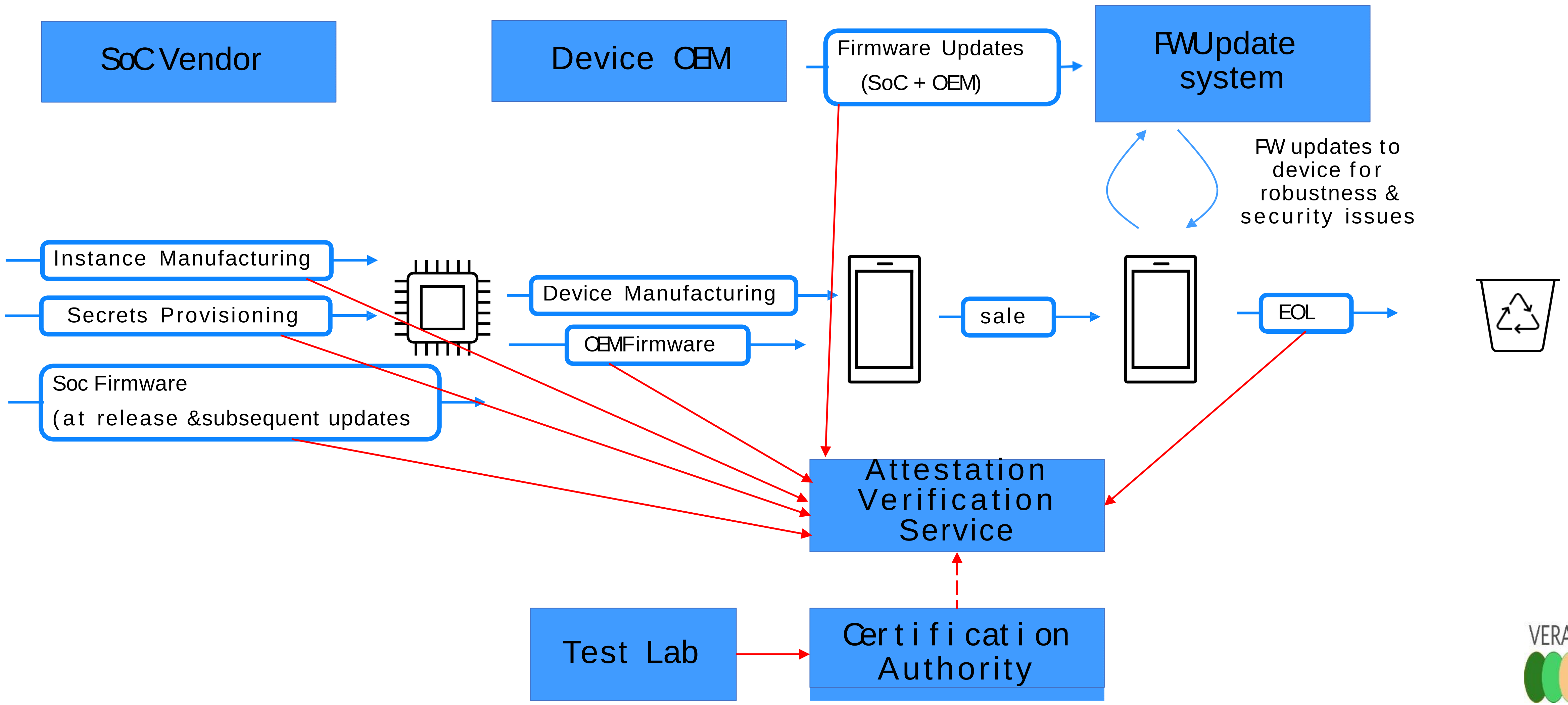
(somewhat idealized)



SoC Vendor
Device OEM
Firmware Updates (SoC + OEM)
FW Update System
FW updates to device for robustness & security issues
Instance Manufacturing
Secrets Provisioning
Soc Firmware (at release &subsequent updates
Device Manufacturing
OEM Firmware
sale
EOL

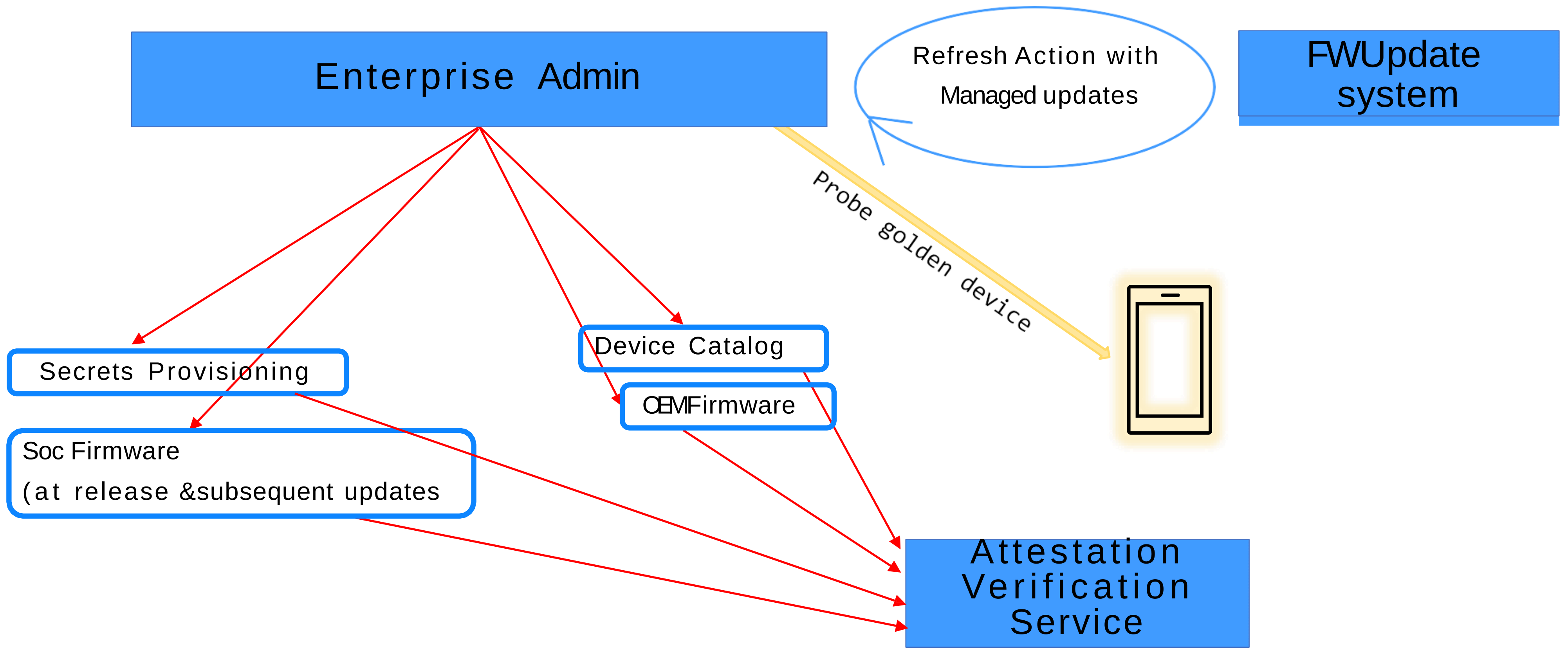
VERAISON

Information Flow for Verification


SoC Vendor
Device OEM
Firmware Updates (SoC + OEM)
FW Update system
FW updates to device for robustness & security issues
Instance Manufacturing
Secrets Provisioning
Soc Firmware (at release &subsequent updates
Device Manufacturing
OEM Firmware
sale
EOL
Attestation Verification Service
Test Lab
Certification Authority

VERAISON

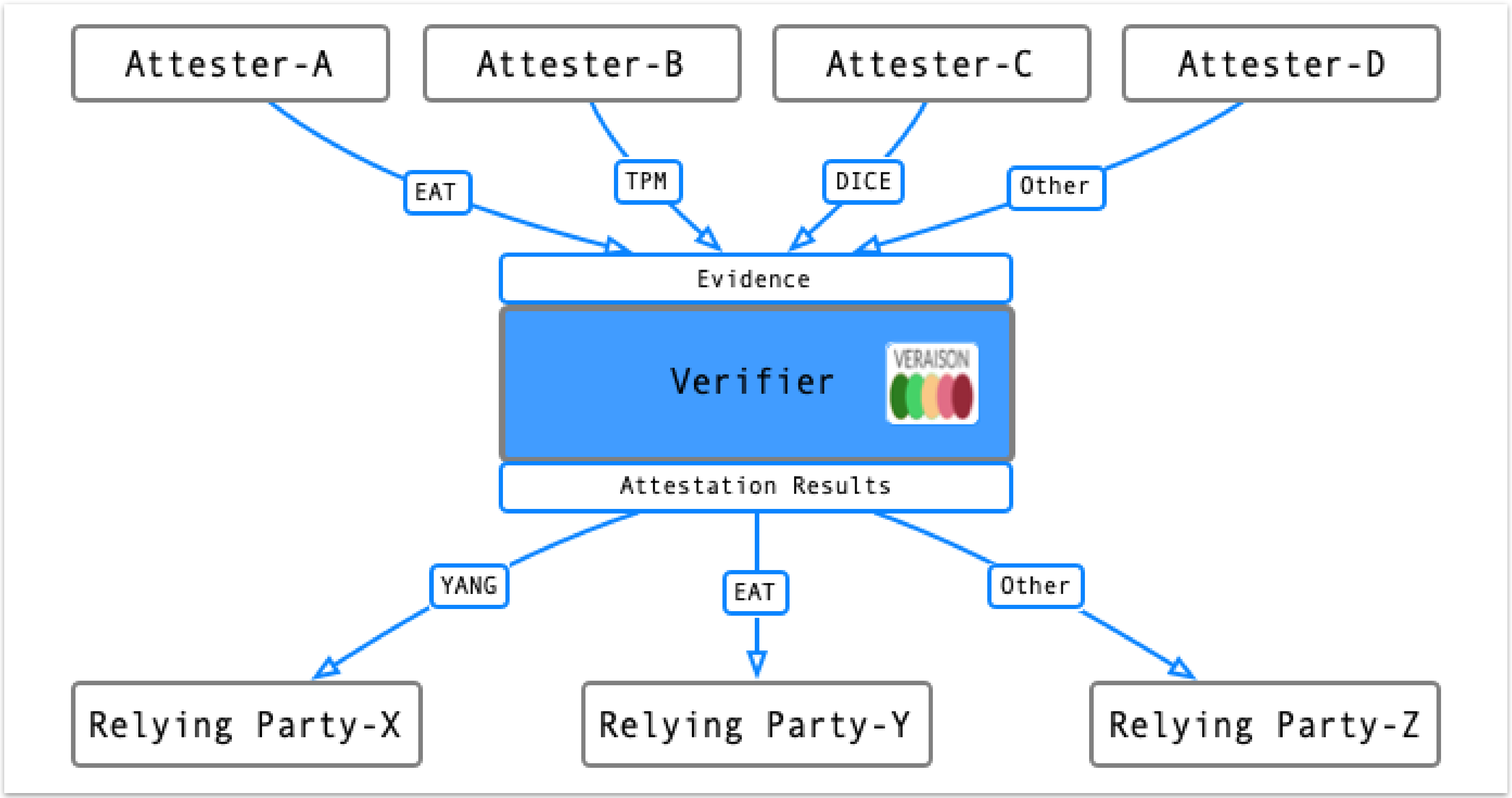
Information Flow for Verification (Enterprise)



Enterprise Admin
Refresh Action with Managed updates
FWUpdate system
Probe golden device
Secrets Provisioning
Device Catalog
OEMFirmware
Soc Firmware (at release &subsequent updates
Attestation Verification Service

VERAISON

Attester and Result Heterogeneity



VERAISON

Project Veraison

- **VER**ific**A**t**I**on of atte**station**
- Open Source (Apache v2.0) & Open Governance
- Collection of libraries and tools for implementing a remote attestation verification service
- A Confidential Computing Consortium project
- Industry wide scope

VERAISON

Design Principles

- Multi architecture
- Model supply chain interaction with Verifier
- Flexible deployment models
  - Public, private, hybrid, multi cloud service
  - Single or multiple tenants
  - Potential to deploy `locally’ e.g. in adjacent isolation such as Trust Zone
- Industry standards used where possible
  - IETF RATS Architecture & Information model
  - TCG DICE Endorsement data format working group

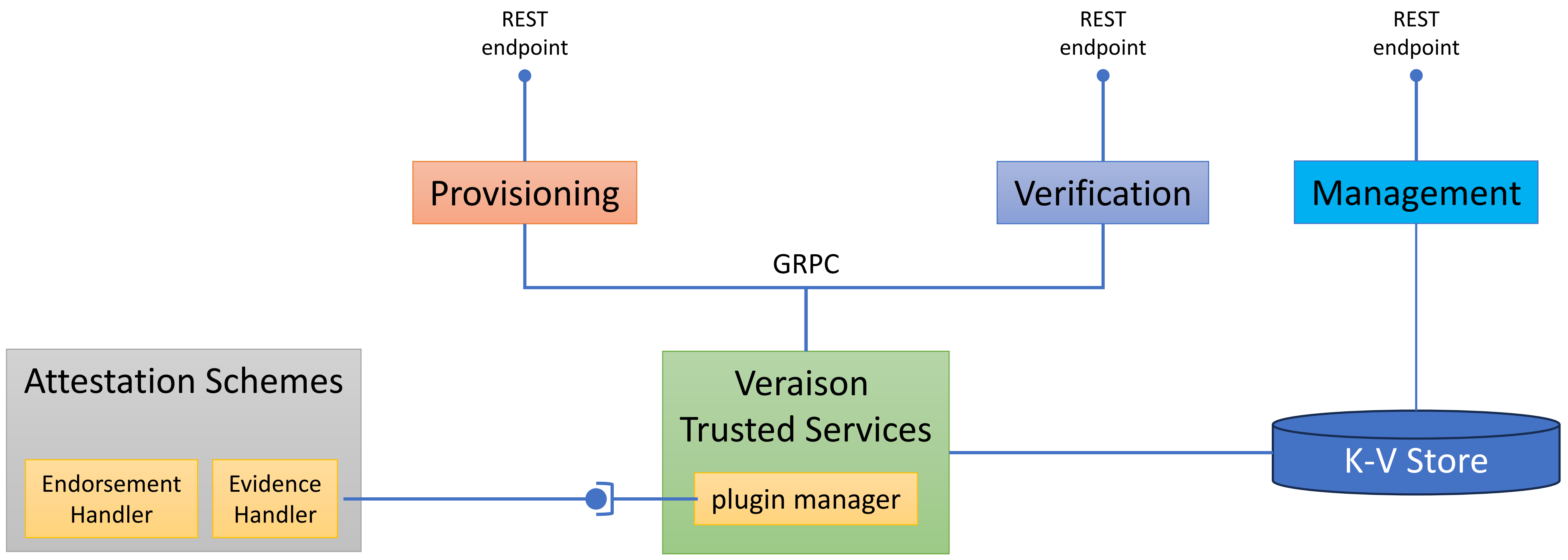
VERAISON

Design Overview

- API Driven
- Support for verification of multiple attestation formats
- Token Verification is flexible
  - Policy driven or extensible via plugins
- Access to Provisioned Reference Values (Endorsements)
- Reference implementations: EAT – PSA Token, Arm CCA, DICE, TPM

Veraison Architecture

Architecture Overview



Provisioning

- Authorised supply chain actors (SoC Vendors, OEM, ISVs etc) need to supply Reference Values & Endorsements to the Verifier

- Veraison uses standards driven Information Model and Data Model to convey Reference Values and Endorsements. This enables:
  - Standard Tooling
  - Reduce Fragmentation
  - Lower barrier to entry for supply chain actors

VERAISON

CoRIM

Concise **R**eference **I**ntegrity **M**anifest

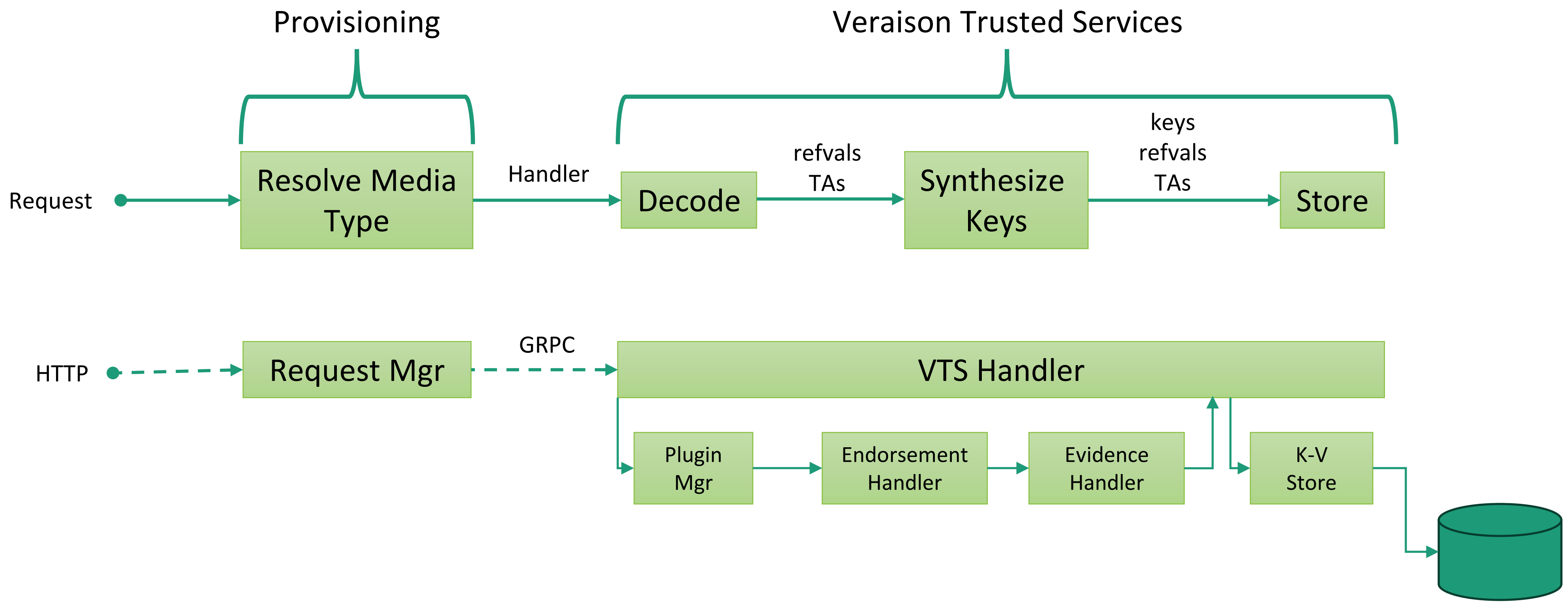
- A signed, **CBOR**-formatted document (**COSE**)
- Data is represented as statements (*i.e. subject-verb-object "triples"*)

component "X" – has reference values – [list of values]

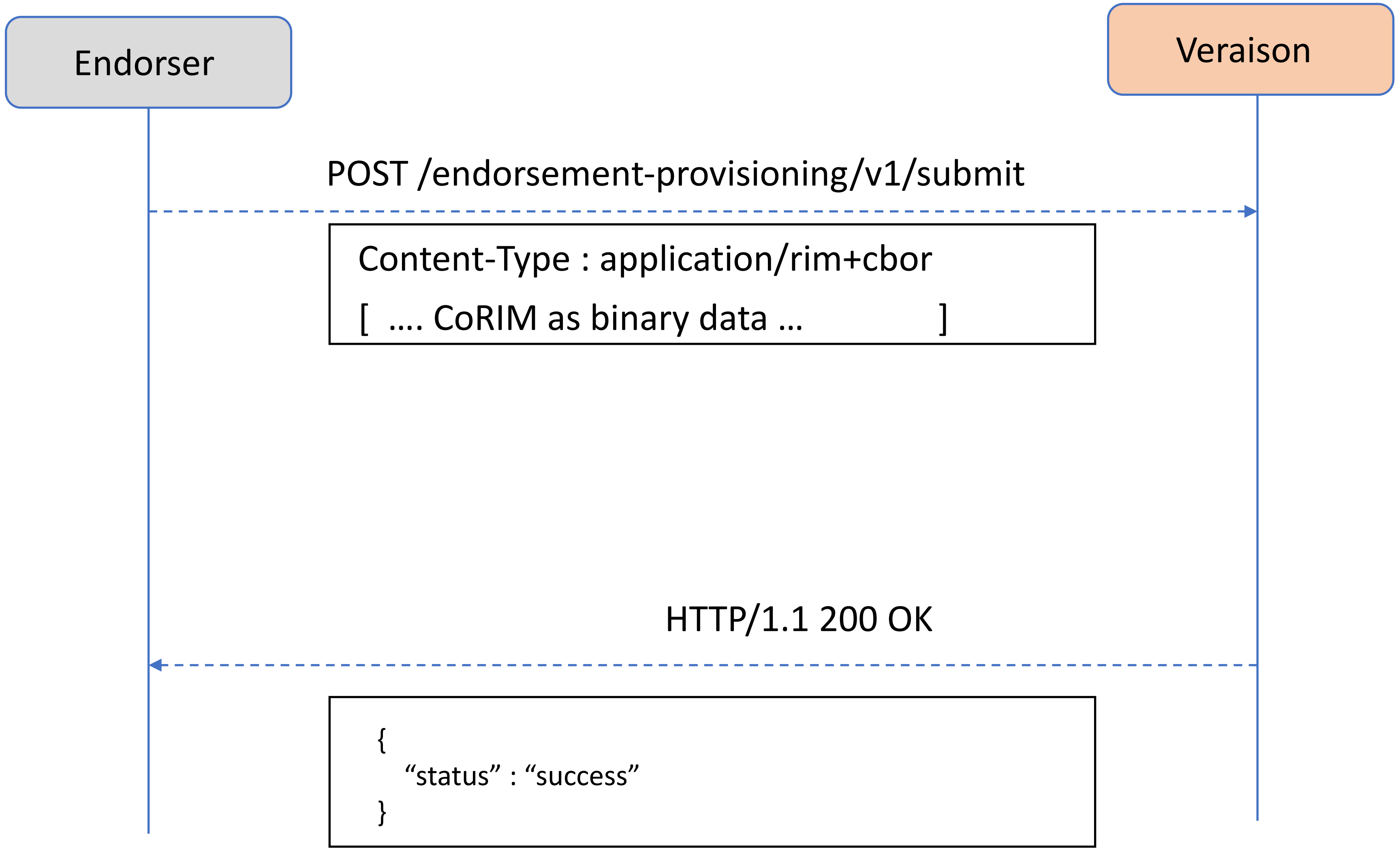
- CoRIM has CoMIDs and CoSWIDs that carry RV and EV from Supply Chain
- Also contains metadata (provisioner identity, versioning etc.)
- Veraison CoRIM is an implementation of CoRIM standards being developed in IETF RATS and TCG working groups
  - ➢ https://datatracker.ietf.org/doc/draft-ietf-rats-corim/
  - ➢ TCG Endorsement Architecture

VERAISON

Provisioning Pipeline



Provisioning



```
Content-Type : application/rim+cbor
[  …. CoRIM as binary data …              ]
```

```
{
   "status" : "success"
}
```

CoRIM Template Excerpt

```
"entities": [{
 "name": "ACME Corp.",
 "regid": "https://acme.com",
 "roles": [ "tagCreator", "creator", "maintainer"]
 }],
 "triples": {
 "reference-values": [
  {
 "environment": { "instance": {"type":"uuid", "value": "7d<...>f1" }},
"measurements": [
 { "value": { "digests": [ "sha-256:h0KPxS<...>MTPJcc=" ] } }
  ]
  }
 }
```

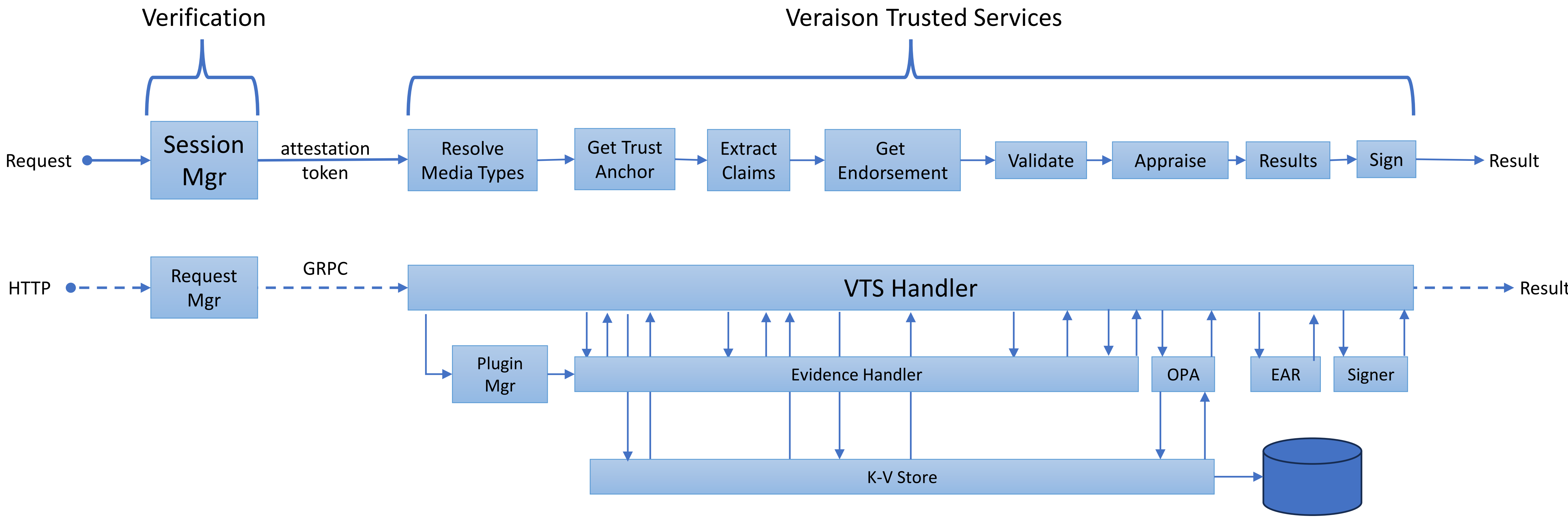
VERAISON

Verification

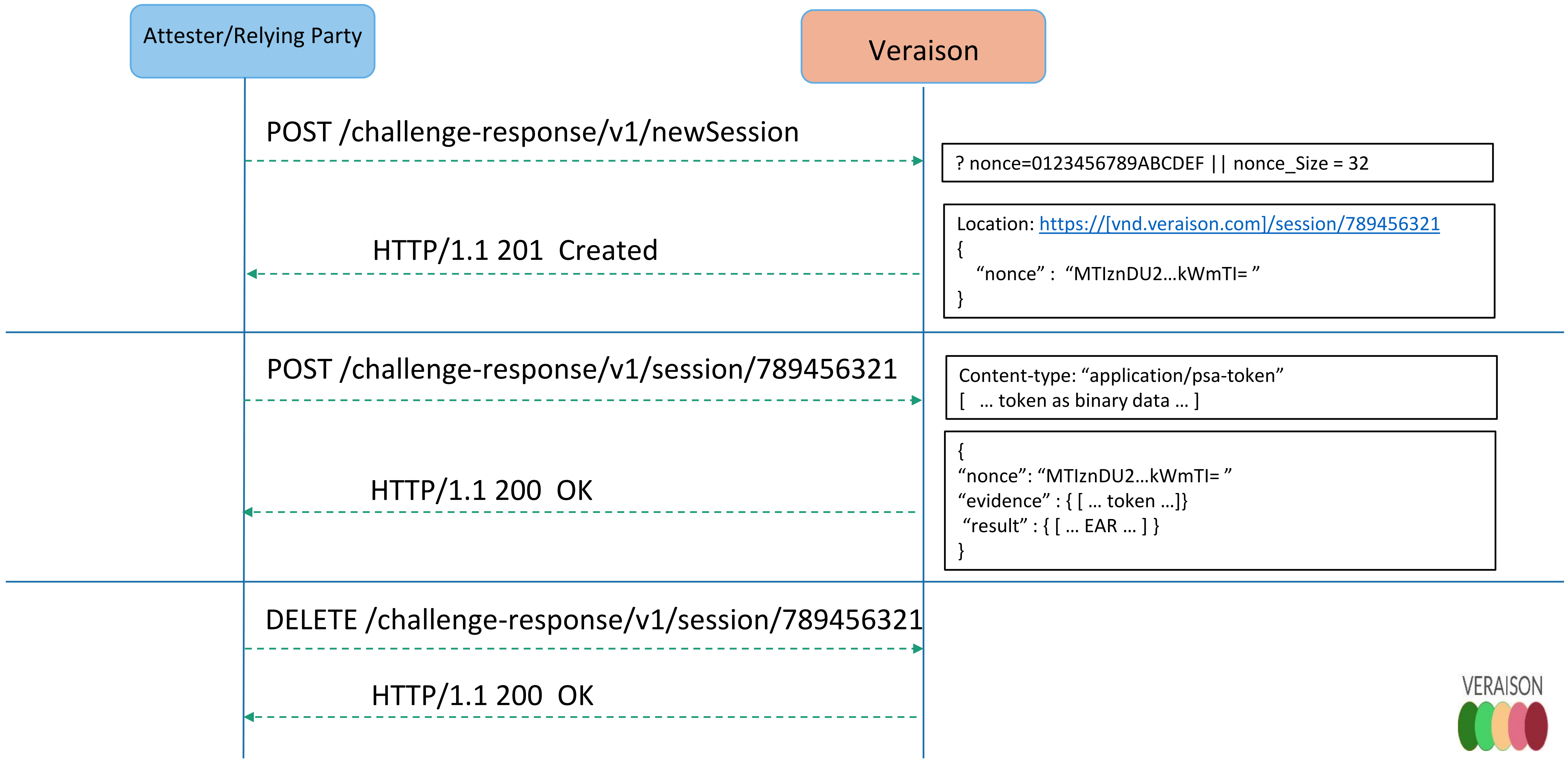
- A session is established with an agreed upon **nonce**
- Attester/Relying Party submits Evidence to the Verification service
- Gets signed **Attestation Results** as an EAR document
- https://github.com/veraison/docs/blob/main/api/challenge-response/README.md

VERAISON

Verification Pipeline



Verification



POST /challenge-response/v1/newSession

```
? nonce=0123456789ABCDEF || nonce_Size = 32
```

HTTP/1.1 201 Created

```
Location: https://[vnd.veraison.com]/session/789456321
{
   "nonce" :  "MTIznDU2...kWmTI= "
}
```

POST /challenge-response/v1/session/789456321

```
Content-type: "application/psa-token"
[   ... token as binary data ... ]
```

HTTP/1.1 200 OK

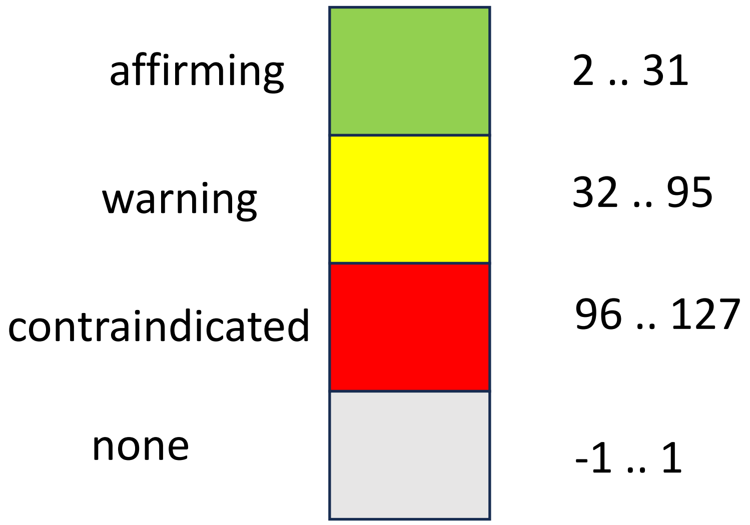
```
{
"nonce": "MTIznDU2...kWmTI= "
"evidence" : { [ ... token ...]}
 "result" : { [ ... EAR ... ] }
}
```

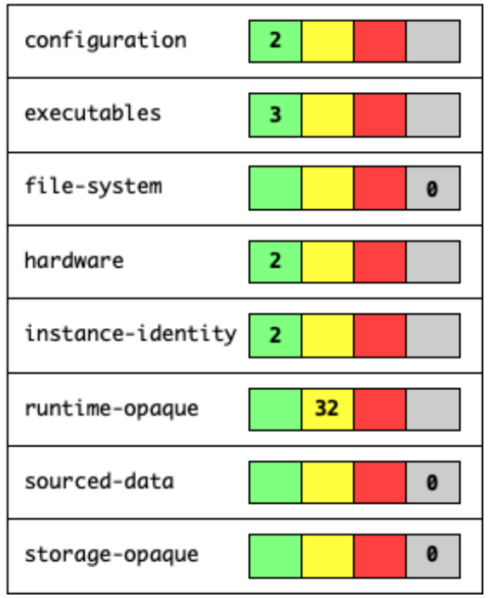
DELETE /challenge-response/v1/session/789456321

HTTP/1.1 200 OK

Attestation Results

- IETF Standard AR4SI defines a Trustworthiness Vector
- A format to represent attestation results in a normalized way, e.g.





	🟩	🟨	🟥	⬜
configuration	2			
executables	3			
file-system				0
hardware	2			
instance-identity	2			
runtime-opaque		32		
sourced-data				0
storage-opaque				0

AR4SI ➔ Attestation Results for Secure Interaction
https://datatracker.ietf.org/doc/draft-ietf-rats-ar4si/

EAR

EAT (Entity Attestation Token) **A**ttestation **R**esults

- A signed JSON Document (JWT) containing:
  - An overall status and an AR4SI Trust Vector
  - Annotated Evidence
  - Policy Claims
  - Time of appraisal
  - Identity of the Verifier

- https://datatracker.ietf.org/doc/draft-fv-rats-ear/

EAR Example

```
{
"ear.status": "affirming",
 "ear.trustworthiness-vector": {
 "configuration": 0,
 "executables": 2,
 [ ... ]
 },
 "ear.veraison.annotated-evidence": {
 "firmware-version": 7,
 "pcr-selection": [1, 2, 3, 4],
 "pcr-digest": "h0KPxSKAPTEGXnvOPPA/5HUJZjHl4Hu9eg/eYMTPJcc=",  [ ... ]
   }
}
```

VERAISON

Attestation Scheme

- Defines:
  - Evidence token structure
  - What Reference Values, Endorsements, and Trust Anchors are expected
  - How the Evidence is appraised

- Implemented via pluggable interfaces

- May be augmented via deployment-specific policies

Policies

- Allow “post-processing” of attestation results generated by scheme
  - ➢ Override Appraisal Decisions
  - ➢ Insert additional claims

- Implemented using **O**pen **P**olicy **A**gent (**OPA**) engine

- Written in **Rego** language

- Policies are handled via Management Interface

VERAISON

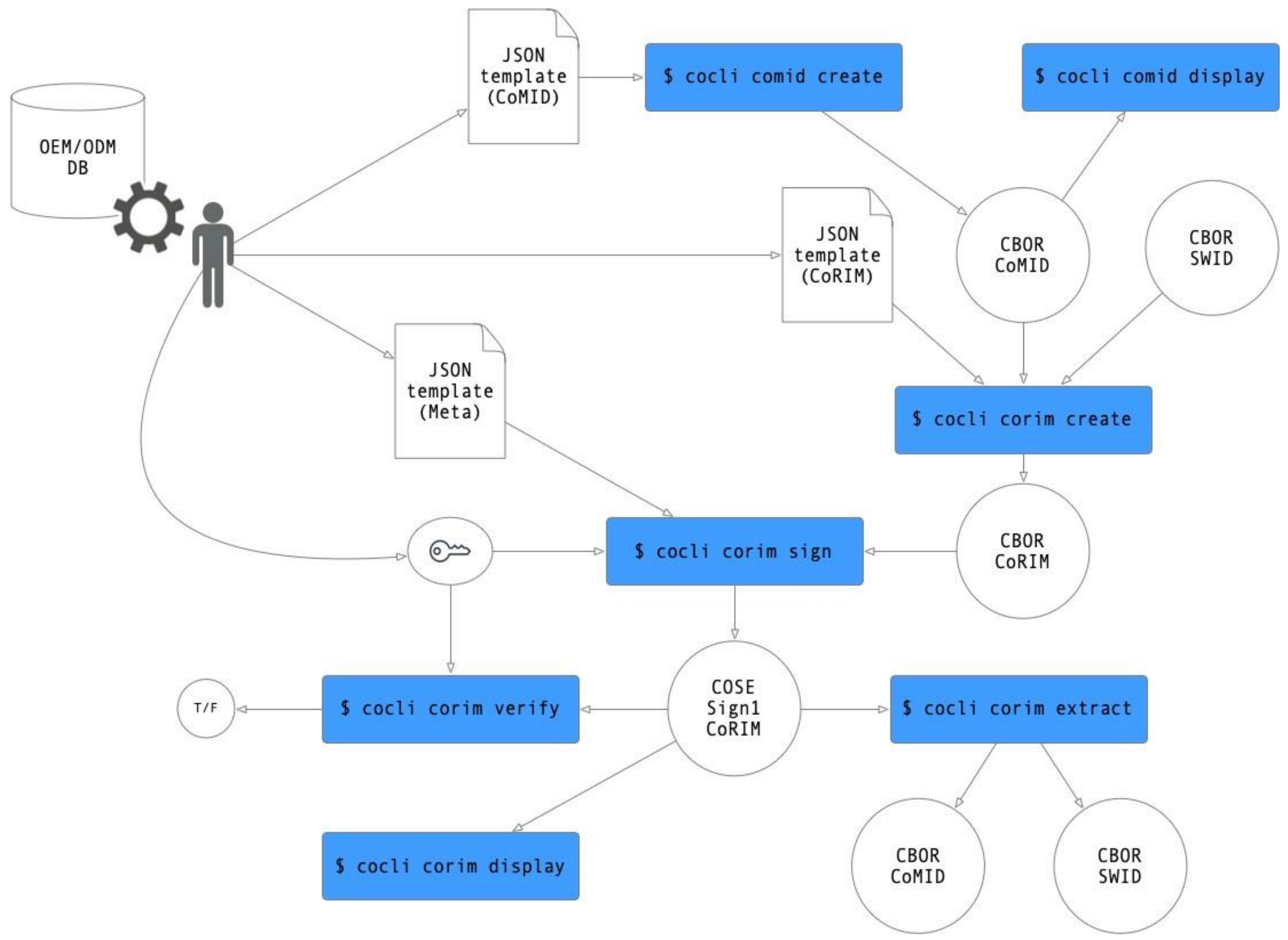
Policy Example

```
# This sets executables trust vector value to AFFIRMING iff BL version is # 3.5 or greater, and to
failure otherwise.
executables = "AFFIRMING" {

# there exists some `I’, such that...
some i

# ...the i'th software component has type "BL", and...
evidence["psa-software-components"][i]["measurement-type"] == "BL"

# … the version of this software component is greater than or equal to 3.5
# (semver_cmp is defined by the policy package. It returns 1 if the first parameter is
# greater than the second, -1 if it is less than the second and 0 if they are equal


semver_cmp(evidence["psa-software-components"][i].version, "3.5") >= 0
} else = "CONTRAINDICATED"  # unless the above condition is bet return “CONTRAINDICATED”
```

Libraries and Tooling

Tooling for the Supply Chain

[veraison/corim/cocli]



VERAISON

Other Tools

Tool	Purpose
evcli	A handy tool to manipulate Evidence to/from CBOR using JSON Claims and a crypto key Also allows exchanging Evidence with Veraison (acting as Attester or Relying Party)
arc	A CLI tool to manipulate Attestation Results
pocli	A CLI tool to manage Policies, i. e. Create, Activate, Deactivate & list Policies for a scheme
gen-corim	A handy CLI tool to generate CoRIM Endorsements from Evidence token

VERAISON

Current Status

- REST APIs for Access to Services
- Support for Multiple Attestation technologies
  - Implemented : PSA , CCA, TPM, DICE { OpenDICE, TCG DICE }
  - Work In Progress (AMD-SEV-SNP)
- Multi-tenancy roles and Authorization support
- Container deployment
- First implementation of standards : CoRIM/EAT Claims + Attestation Results
- Support for CoRIM Extensions – for multiple schemes { TDX, AMD-SEV-SNP}
- Deployable appraisal policy
- PoC Deployment `in TEE` with proofs

On the Roadmap

- Options to deploy without (external) plugin framework to reduce TCB
- Support for further Attestation Architectures – e.g. Intel TDX
- Inline Endorsements
- Support for Event Logs
- Exploring constrained deployment in local TEE

VERAISON

Out of Scope

- It is not intended to look at other aspects of verification e.g.
  - Unification of Attestation Token formats
  - Normalising how a Relying Party requests Attestation
  - Common Attestation protocol

VERAISON

Get Involved!

- We would be very interested in collaboration from this skilled and knowledgeable community
  - Principles/Assumptions
  - Design Aspects
  - Extend Veraison to support a new scheme to match the use case
  - Consumption/Reference deployments

- Join us on Zulip at https://veraison.zulipchat.com/

- Welcome to discuss @ Weekly Community Meet
  - Every Tuesday 4PM UK time

VERAISON

VERAISON

https://github.com/veraison/