Project Veraison

Attestation Verification Components

Veraison: **VER**ific**A**t**I**on of atte**S**tati**ON**

Setting the scene:

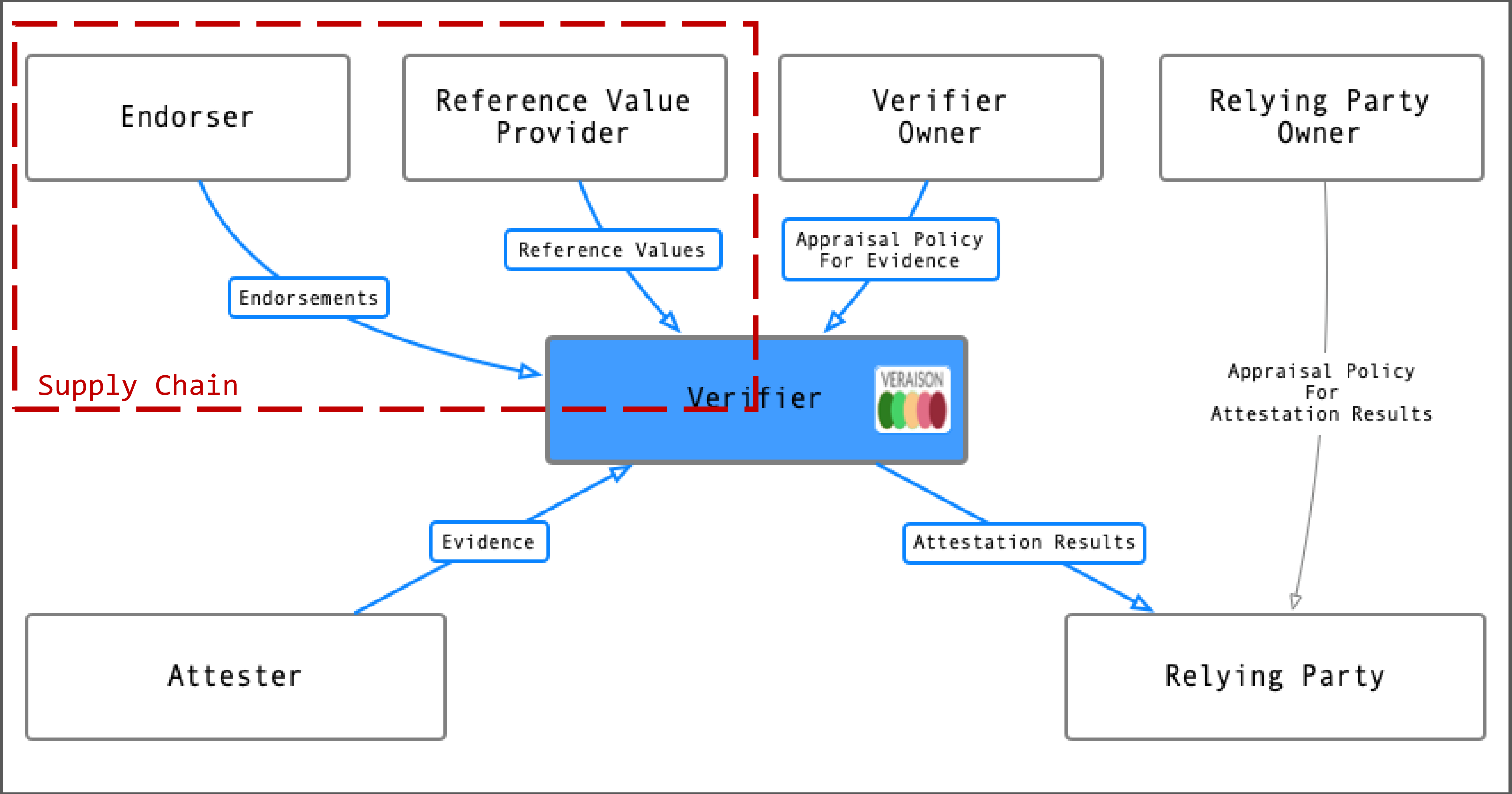
- *“Confidential Computing protects data in use by performing computation in a hardware-based Trusted Execution Environment”*
  - *Confidential Computing: Hardware-Based Trusted Execution for Applications and Data*
- CC service users *must* be able to establish that a TEE is trustworthy
  - Hardware & Software aspects are "correct"
- The means to establish trustworthiness is Attestation
- Being able to produce an Attestation report alone is not sufficient
- The report must be Verified to prove the constituent claims

VERAISON

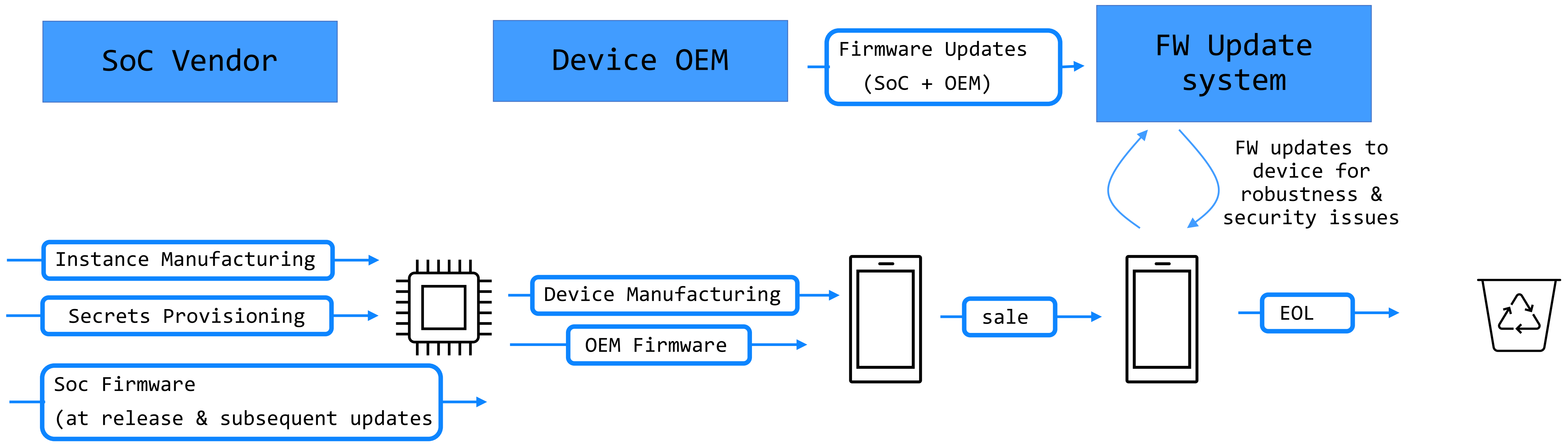
Process of Verification

- Verification operations must:
  - Deserialise & syntax check attestation report data models
  - Check Cryptographic Signing
    - which requires knowledge of relevant trust anchor(s)
  - Confirm measurements within claims match Reference Values
    - Ref values need to be drawn from supply chain
    - and be up to date
    - Multiple actors, business and trust relationships
  - Apply any semantic relationships between claims
    - e.g., certain hardware & firmware combinations
- Perform all operations while being trustworthy itself

VERAISON

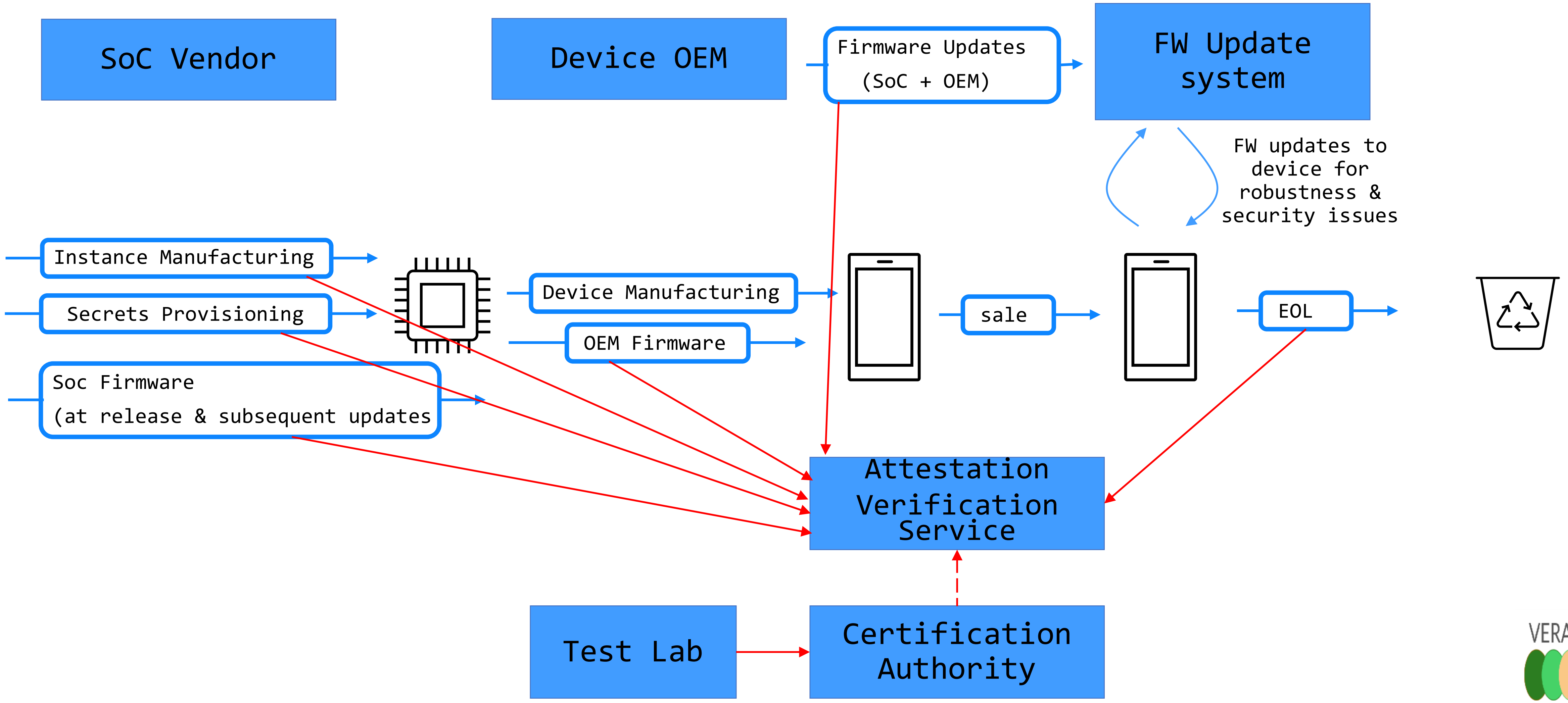

Endorser
Reference Value Provider
Verifier Owner
Relying Party Owner
Endorsements
Reference Values
Appraisal Policy For Evidence
Supply Chain
Verifier
VERAISON
Appraisal Policy For Attestation Results
Evidence
Attestation Results
Attester
Relying Party

Supply Chain & Lifecycle (somewhat idealised)



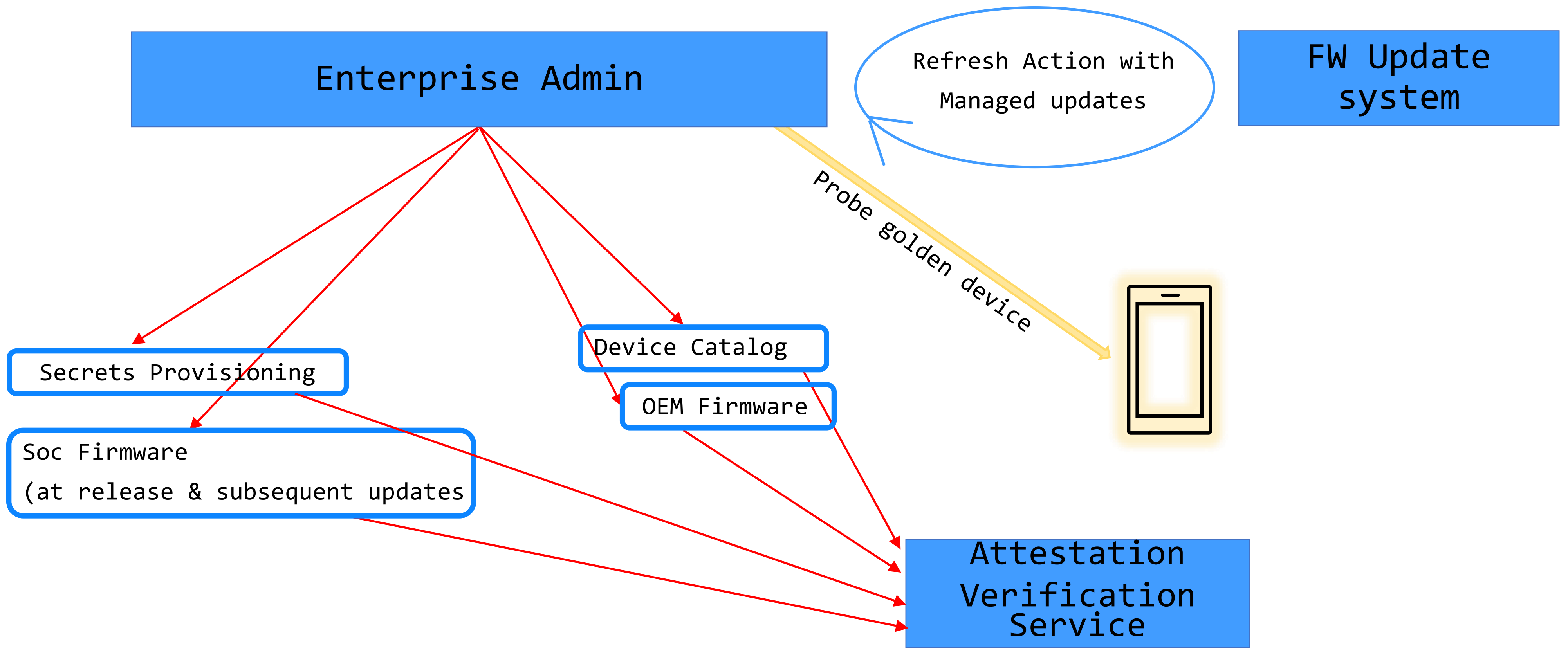
VERAISON

Information Flow for Verification

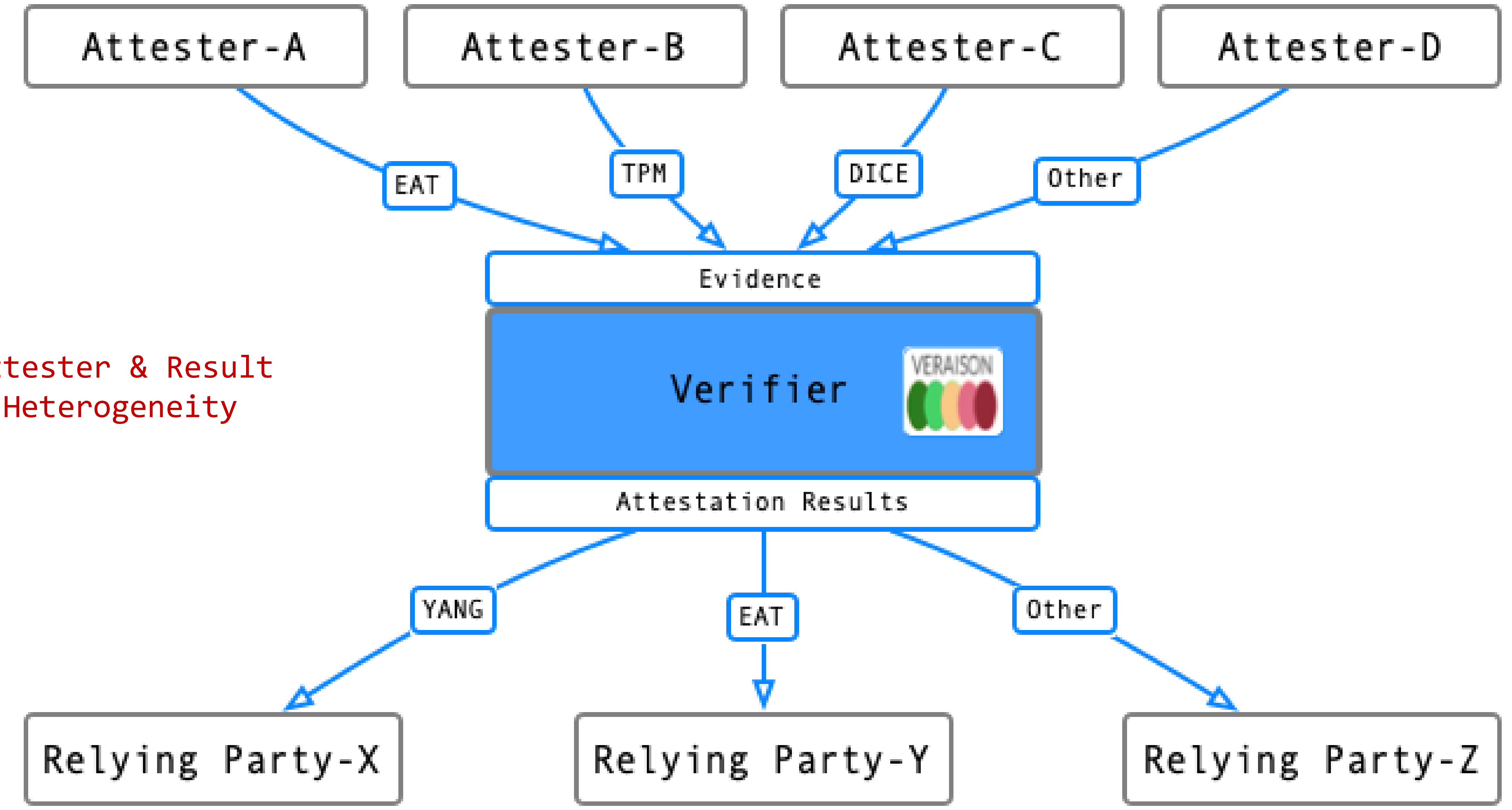

SoC Vendor
Device OEM
Firmware Updates (SoC + OEM)
FW Update system
FW updates to device for robustness & security issues
Instance Manufacturing
Secrets Provisioning
Soc Firmware (at release & subsequent updates
Device Manufacturing
OEM Firmware
sale
EOL
Attestation Verification Service
Test Lab
Certification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

VERAISON

Information Flow for Verification (Enterprise)


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

VERAISON



Attester & Result
Heterogeneity

Verification software components

- If every deployment prepares custom logic for the verification process
  - Quality, and hence trustworthiness, may vary
  - Building a verifier is a barrier to entry
- Project Veraison (**VER**ific**A**t**I**on of atte**S**tati**ON**) will build software components that can be used for attestation verification services
- Open-source project, operating with fully Open Governance
- Arm is making contributions to the core team, but the intent is to have an industry wide scope

VERAISON

The why of Veraison

- Observation:
  - verification is at the centre of virtually any attestation flow
- To enable attestation-based protocols *pervasively*, verification must become:
  - as {ubiquitous, flexible, reusable, composable, secure, open} as possible
- i.e., verification needs to be commodified
- Veraison aims at becoming a building block in the commoditisation of verification

Target use cases

- Flexible deployment models
  - public, private, hybrid, multi cloud service
  - Narrow / wide in terms of supported attestation formats
  - Single or multiple tenants

- Remote vs Local
  - Remote verification is the normal pattern
  - Veraison software components could be deployed on a local basis
    - especially in isolated execution environment such as TrustZone

VERAISON

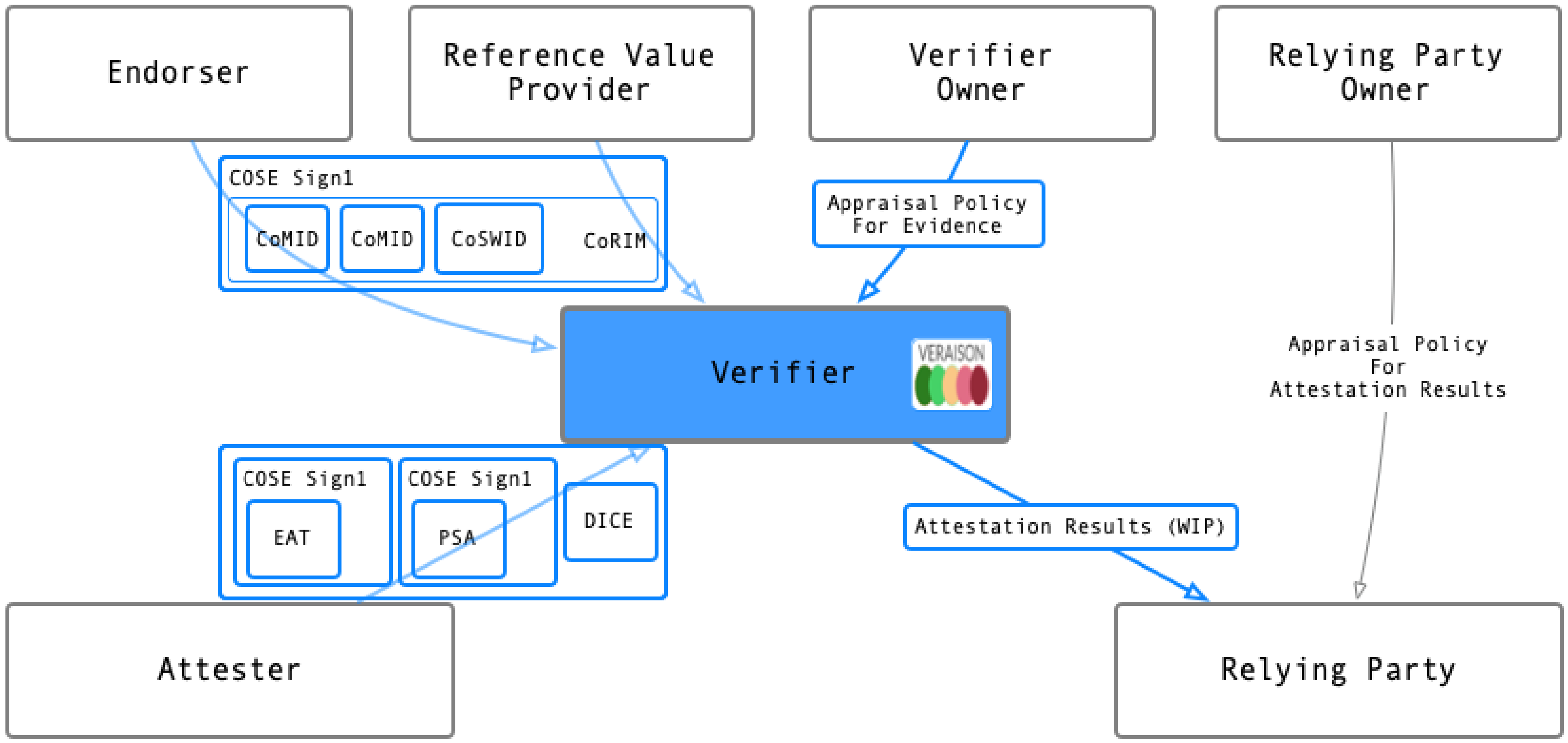
Veraison Software Architecture

Veraison deliverables fall into two categories:

- Attestation data formats
  - Endorsed and Reference values
  - Trust Anchors
  - Evidence
  - Attestation results
- Verifier pipeline(s)
  - Verification
  - Provisioning (of endorsed and reference values, and trust anchors)

VERAISON

Attestation data formats



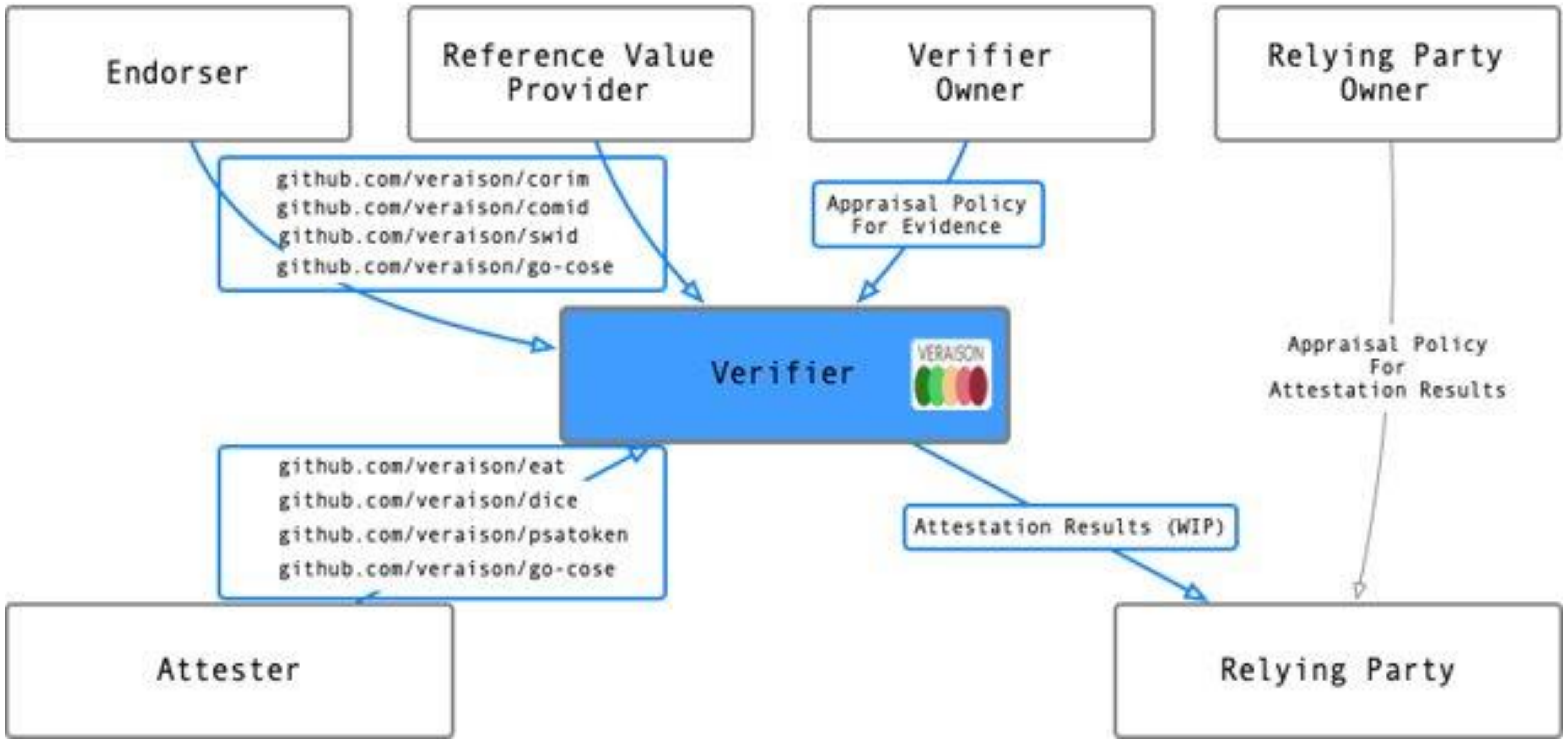
VERAISON

Trust vector

- A format to represent attestation results in a normalised way
- Checklist of typical micro-appraisals, e.g.:
  - Attester Identity
  - TCB integrity

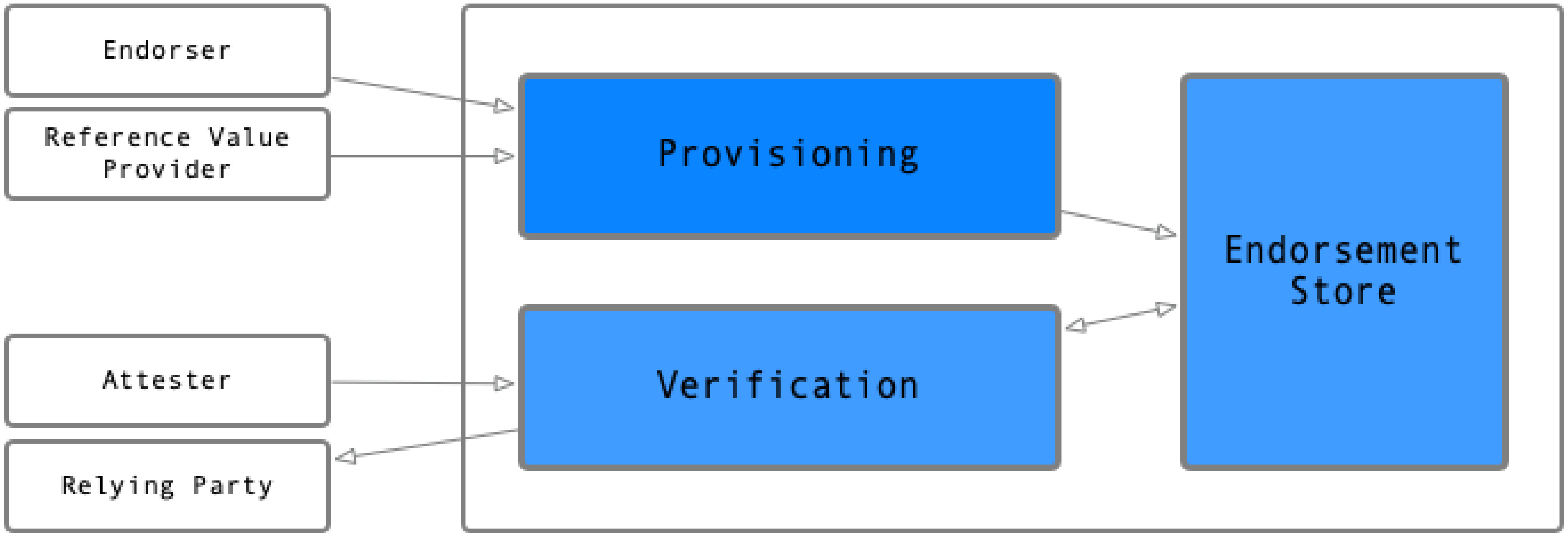
- Significant step in the standardisation of attestation formats, in particular WRT relying parties

Attestation data formats


Endorser
Reference Value Provider
Verifier Owner
Relying Party Owner
github.com/veraison/corim
github.com/veraison/comid
github.com/veraison/swid
github.com/veraison/go-cose
Appraisal Policy For Evidence
Verifier
VERAISON
Appraisal Policy For Attestation Results
github.com/veraison/eat
github.com/veraison/dice
github.com/veraison/psatoken
github.com/veraison/go-cose
Attestation Results (WIP)
Attester
Relying Party

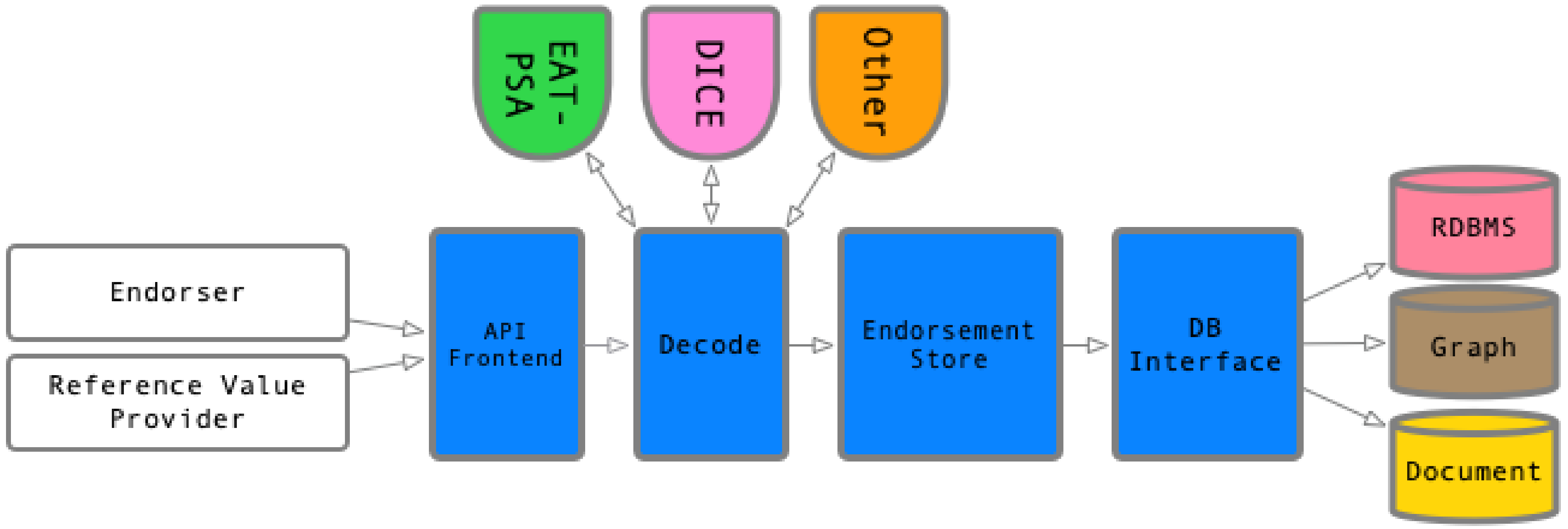
VERAISON

Verifier pipelines



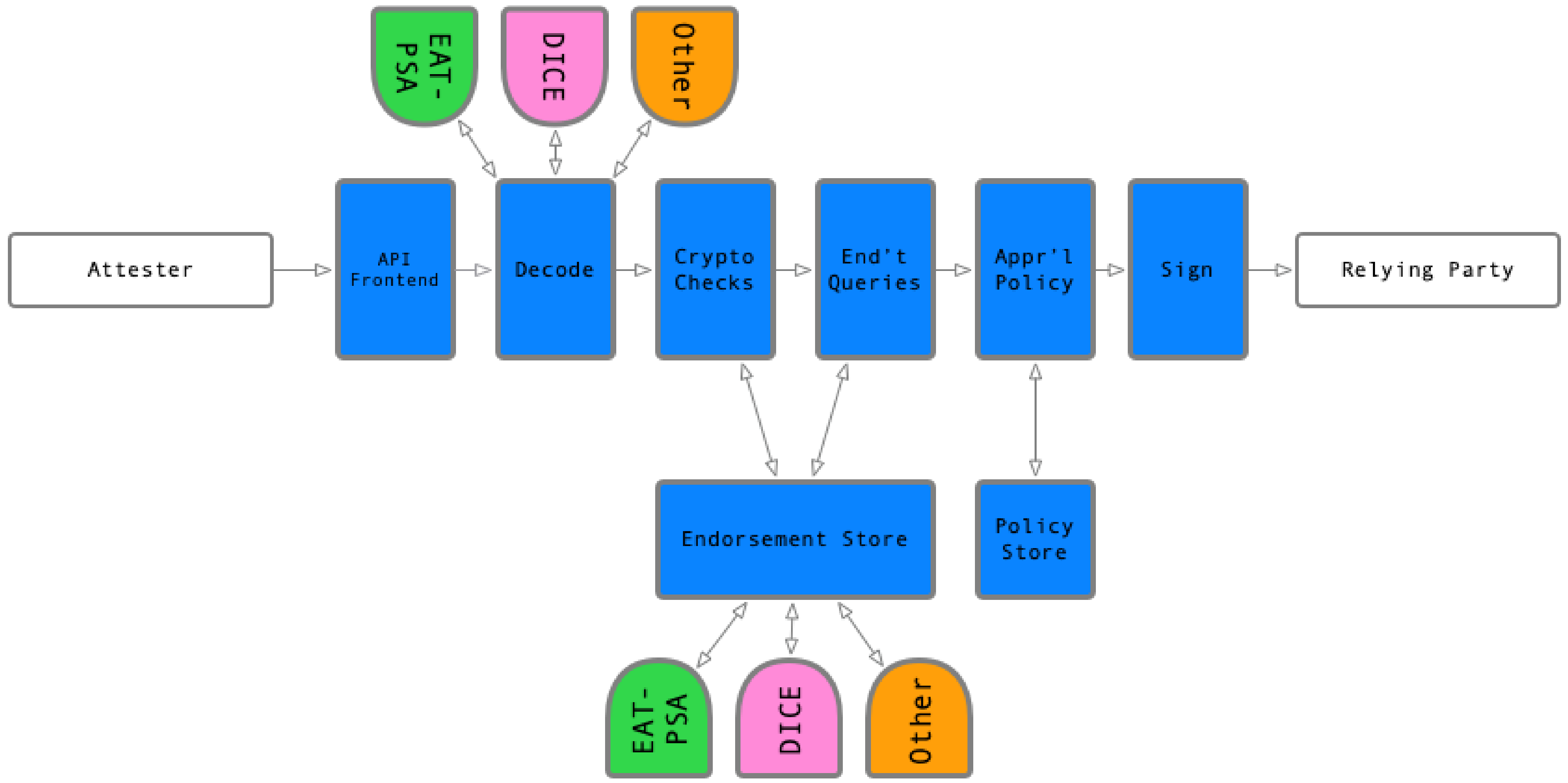
VERAISON

Provisioning pipeline



VERAISON

Verification pipeline



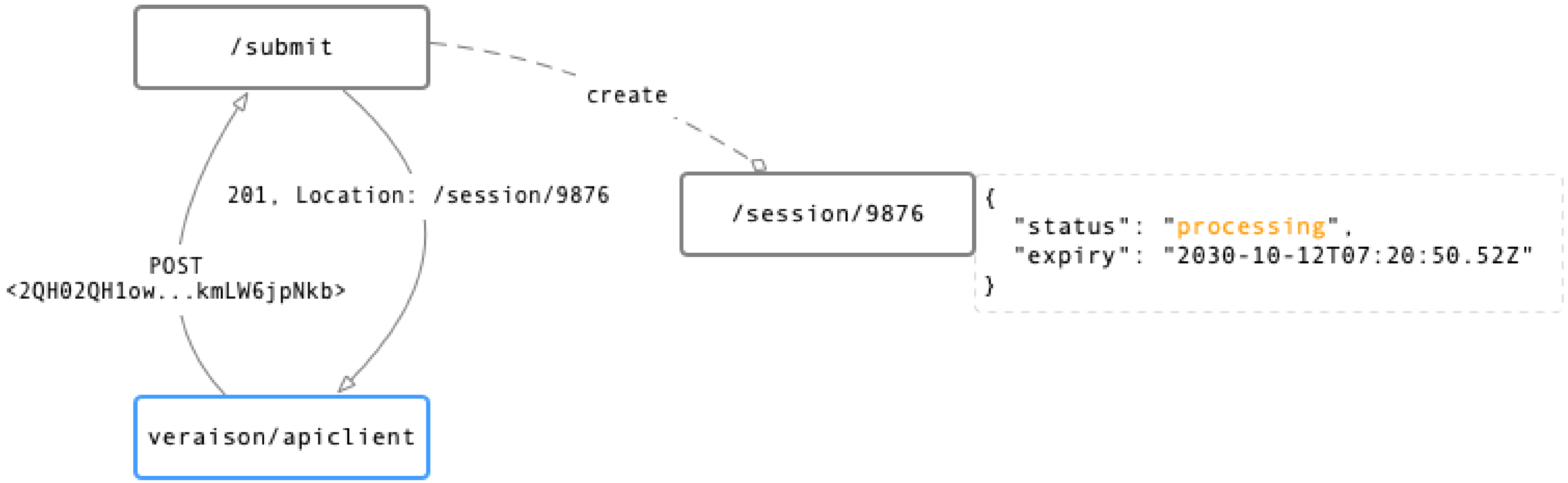
VERAISON

Veraison REST API

- Provisioning
    - Submit endorsements
- Verification
    - Challenge-response session
- Server
    - `github.com/veraison/veraison/{provisioning,verification}`
- Client
    - `github.com/veraison/apiclient`

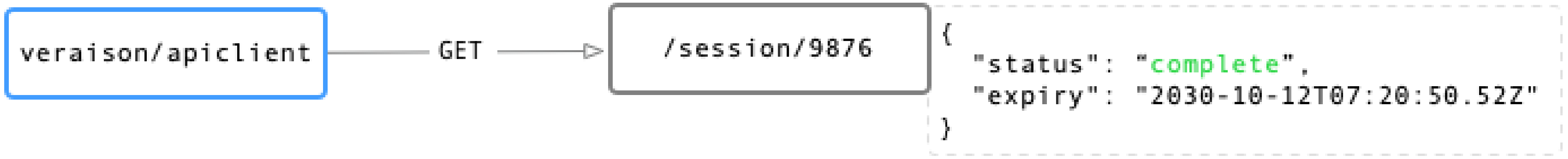
VERAISON

Veraison REST API, provisioning (1)



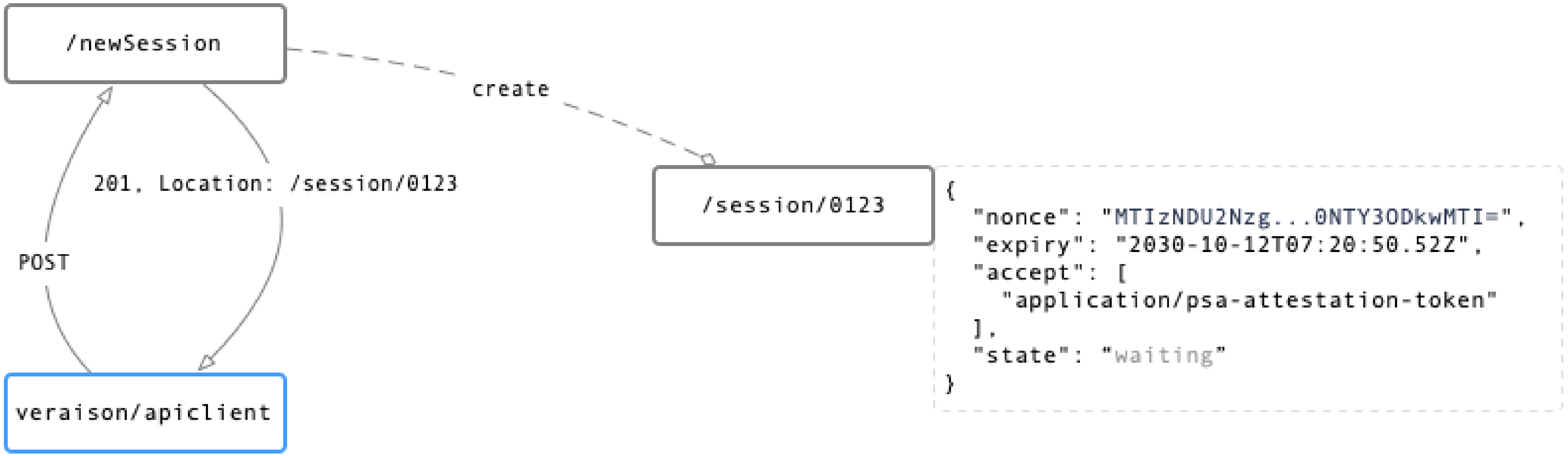
```
{
  "status": "processing",
  "expiry": "2030-10-12T07:20:50.52Z"
}
```

Veraison REST API, provisioning (2)



```
{
  "status": "complete",
  "expiry": "2030-10-12T07:20:50.52Z"
}
```

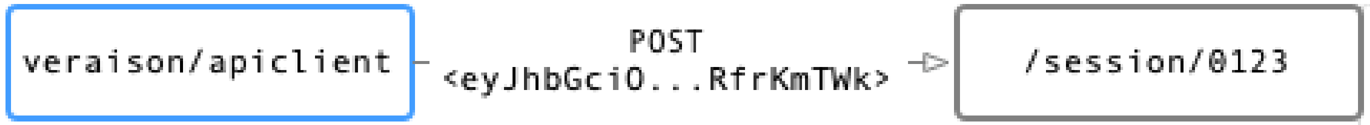
Veraison REST API, verification (1)



```
{
  "nonce": "MTIzNDU2Nzg...0NTY3ODkwMTI=",
  "expiry": "2030-10-12T07:20:50.52Z",
  "accept": [
    "application/psa-attestation-token"
  ],
  "state": "waiting"
}
```

VERAISON

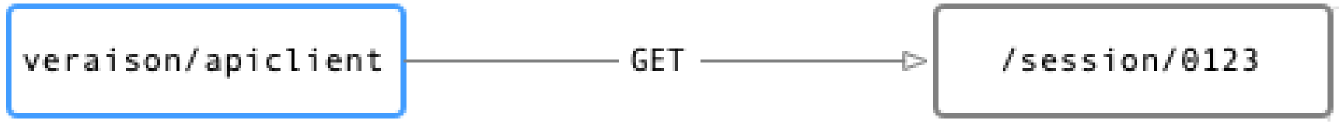
Veraison REST API, verification (2)



```
{
  "nonce": "MTIzNDU2Nzg...ONTY3ODkwMTI=",
  "expiry": "2030-10-12T07:20:50.52Z",
  "accept": [
    "application/psa-attestation-token"
  ],
  "state": "processing",
  "evidence": {
    "type": "application/psa-attestation-token",
    "value": "eyJhbGciO...RfrKmTWk"
  }
}
```

VERAISON

Veraison REST API, verification (3)



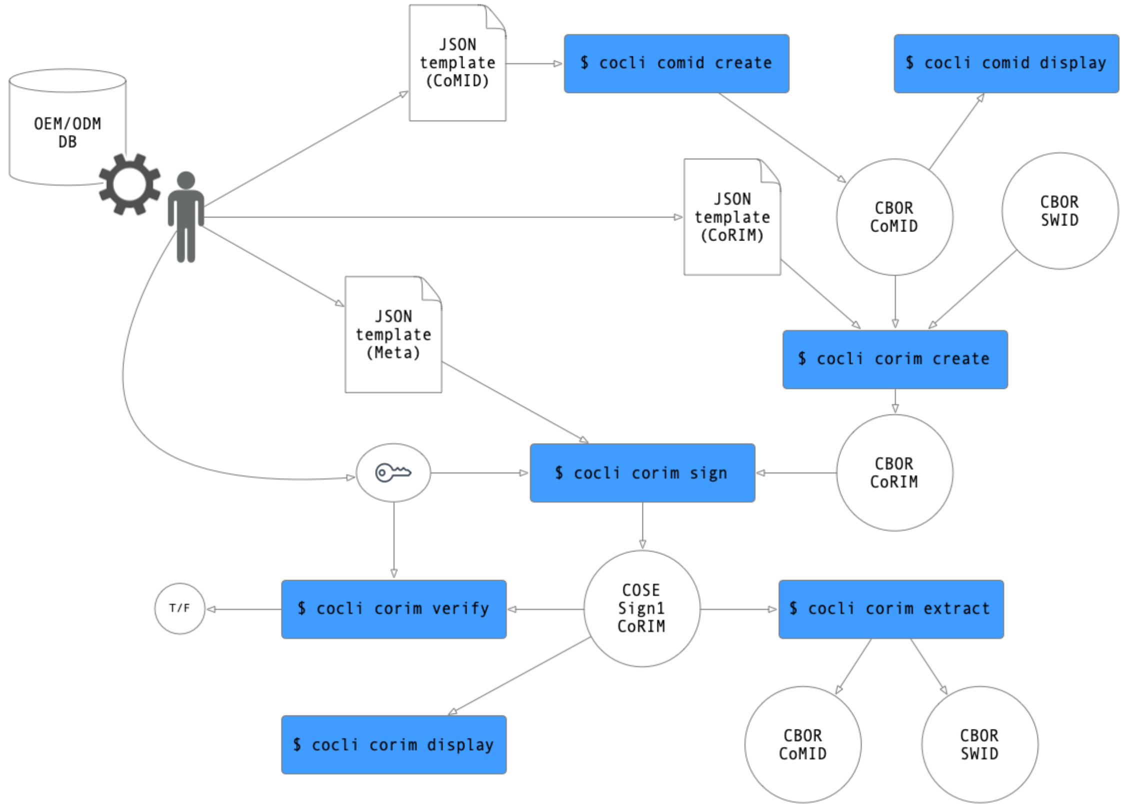
Click to add text

```
{
  "nonce": "MTIzNDU2Nzg...ONTY3ODkwMTI=",
  "expiry": "2030-10-12T07:20:50.52Z",
  "accept": [
    "application/psa-attestation-token"
  ],
  "state": "complete",
  "evidence": {
    "type": "application/psa-attestation-token",
    "value": "eyJhbGciO...RfrKmTWk"
  },
  "attestation_results": {
    "valid": true,
    // ...
  }
}
```

VERAISON

Tooling for the supply chain [veraison/corim/cocli]

- Challenge to supply chain practices
- Embracing standards
- Provide tooling that integrates with the production events



VERAISON

Turning components into a service

- Components use abstraction over underlying platform services
  - e.g., Data Storage
- Reference implementations for Container + a (TBD) Cloud
- Basic implementations for features such as
  - Audit trail
  - Multi tenancy isolation of provisioned data
- Scaling exploits concept of ‘Compute Units’
  - Veraison components built for discrete operation
  - Components will instrument their usage
  - Build scaling operations on top of such instrumentation

VERAISON

Status

- Project is active on github (https://github.com/veraison)
- Open Governance / weekly public meetings
- Active collaboration with standards bodies (IETF RATS, TCG)
- Initial Design work complete
- Core functionality demostones (PSA-EAT, DICE)
- Long list of potential features / capabilities
- Contributions most welcome

VERAISON

VERAISON

https://github.com/veraison/