24 July 2026

IETF 126 Session title

This session is being recorded

IETF®

Note Well

By participating in the IETF you agree to follow IETF processes and policies. This Note Well is a reminder of some of those policies. For a linked version of this text, please visit **www.ietf.org/note-well** or use the QR code below

- IETF participants are expected to behave in a professional manner and extend respect and courtesy to their colleagues at all times (see **RFC 7154: IETF Guidelines for Conduct and IETF Anti-Harassment Policy**). If you have any concerns about behavior, please contact the *Ombudsteam* who have a duty of confidentiality and extensive powers to act, as set out in **RFC 7776: IETF Anti-Harassment Procedures**.
- If you are aware that any IETF contribution (as defined in **RFC 5378: Rights Contributors Provide to the IETF Trust**) is covered by patents or patent applications that are owned or controlled by you, your employer or your sponsor, you must disclose that fact, or not participate in the discussion (see **RFC 8179: Intellectual Property Rights in IETF Technology**).
- For detailed process information consult **RFC 2026: Internet Standards Process** and *RFC 2418:* **IETF Working Group Guidelines and Procedures** and updates to those.
- The IETF routinely makes public written, audio, video, and photographic records of IETF activities, including your personal information as set out in the **IETF Privacy Statement**.

For advice, please talk to Working Group chairs or Area Directors.

IETF

This session is being recorded

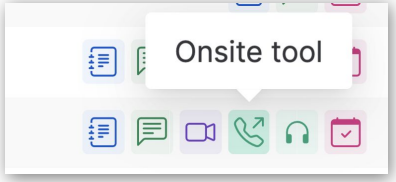
IETF 126 Meeting Tips

In-person participants

- Make sure to sign into the session via Datatracker or the QR Code in this session.
- Use Meetecho (usually the “Meetecho lite”) client to join the queue, show hands
- *Keep audio and video off if not using the onsite version.*

Remote participants

- Make sure your audio and video are off unless you are chairing or presenting during a session.
- Use of a headset is strongly recommended.



All participants

- State your name each time you begin speaking in the queue

Resources for IETF 126 Vienna

- Agenda
  https://datatracker.ietf.org/meeting/agenda
- Meetecho and other information:
  https://www.ietf.org/meeting/preparation/
- If you need technical assistance, see the Reporting Issues page:
  https://www.ietf.org/meeting/issues/

Agenda

- Introduction (Chairs) 5 min
- Update on SCITT WG specs (Henk Birkholz) 5 minutes
- CCF Profile and Joint Statements extension (Amaury Chamayou) 10 Minutes
- MMR Profile - Call for WG Adoption - (Jon Geater) 10 Minutes
- Hackathon Report (Steven Mih) 12 Minutes
- New use cases incoming (Anton & Evangelos & Gregor) 5+5+5 mInutes
- Future of the Working Group Discussion (Chairs) 23 Minutes
    a. Scene setting and status (Jon) 3 Minutes
    b. Future options + polls (Chairs) 10 Minutes
    c. Discussions and refinement (All) 10 minutes

SCITT WG Spec Update - Henk Birkholz

RFC 9943 needs one focused erratum for its EDN examples

Five non-normative samples are syntactically invalid; normative CDDL and protocol behavior are unchanged.

01 CDDL nil leaked into EDN

FIGURES 4 • 8 • 9

CURRENT

```
nil,    / Detached payload /
```

CORRECT

```
null,   / Detached payload /
```

CDDL `bstr / nil` is correct. EDN spells the CBOR value null.

02 EDN text strings are unquoted

FIGURES 5 • 10

CURRENT

```
application/example+json
software.vendor.example
transparency.vendor.example
vendor.product.example
```

CORRECT

```
"application/example+json"
"software.vendor.example"
"transparency.vendor.example"
"vendor.product.example"
```

Figure 5: first, second, fourth • Figure 10: third, fourth

SCOPE	IMPACT	SEMANTICS
Non-normative EDN samples	Copy/paste examples fail parsers	No wire-format or behavior change

SCITT WG ACTION **Submit one RFC Editor erratum correcting Figures 4, 5, 8, 9, and 10.**

Sources RFC 9943 • EDN (RFC 8610, App. G) • RFC Editor errata

SCITT • REFERENCE API

SCRAPI: the interoperable HTTP API for SCITT transparency

draft-ietf-scitt-scrapi-11 • Proposed Standard • the mandatory resources needed to register and verify

01 DISCOVER KEYS

GET /.well-known/scitt-keys

Fetch the service's COSE verification keys—either as a key set or by key identifier.

COSE Key Set | application/cbor

02 REGISTER STATEMENT

POST /entries

Submit a COSE Signed Statement. The service returns a receipt now—or a stable location to poll.

201 receipt | 202 + Location

03 RESOLVE RECEIPT

GET /entries/{id}

Resolve the resulting Receipt. A running registration stays on the same resource.

200 receipt | 204 + Retry-After

ONE INTEROPERABLE CONTRACT

COSE + CBOR | Concise Problem Details | 429 + Retry-After | HTTP binding

Out of scope by design: internal VDS mechanics • registration-policy logic • object formats • non-HTTP transports

Sources draft • history • IETF 125

IETF 125 → PUBLICATION

Since IETF 125: feedback turned into protocol clarity

Four published revision cycles took -07 from AD review to an approved -11 in the RFC Editor stream.

14–20 MAR	27 MAR–23 APR	13–21 MAY	26–29 JUN	1–6 JUL
IETF 125 baseline	**Last Call + IANA**	**IESG iteration**	**Discuss → approval**	**RFC Editor**
-07 • AD Evaluation Revised I-D Needed	-08 / -09 • reviews Well-known URI approved	-10 • feedback folded in Revised I-D Needed	-11 • async flow fixed 3 DISCUSS positions cleared	In Progress IANA: RFC-Ed-Ack

4	published revisions
5	completed formal review reports
3	blocking DISCUSS positions cleared
~3.4k	plaintext line add/delete churn*

NOW **RFC EDITOR QUEUE** — IN PROGRESS

*Successive -07→-11 text diffs; includes pagination and line reflow.

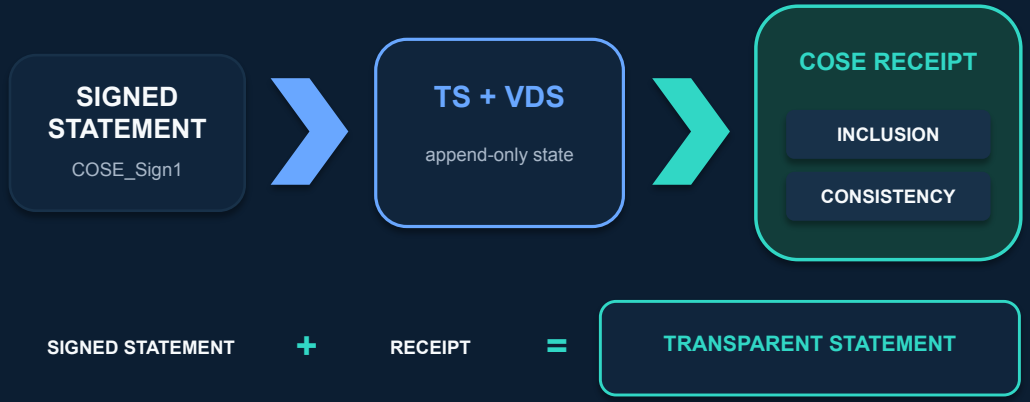
Sources draft • history • IETF 125

COSE • RECEIPTS

RFC 9942: portable proofs for transparency systems

COSE Receipts • Standards Track • became an RFC on 30 Jun 2026

WHAT IT STANDARDIZES



A signed, portable claim about VDS state

CORE COVERAGE

01 Unified receipt format

COSE_Sign1 carries signed proofs about a Verifiable Data Structure.

02 Extensible registries

VDS algorithms and proof types can evolve without changing the envelope.

03 Concrete Merkle profile

Defines RFC9162_SHA256 inclusion and consistency proof encodings.

394 receipts | 395 vds | 396 vdp

WHY SCITT CARES

RFC 9943 requires Transparency Services to produce COSE Receipts. A Receipt proves registration; SCRAPI returns it, and attaching it creates a Transparent Statement.

Sources RFC 9942 • publication history • SCITT architecture

COSE • HASH ENVELOPE

RFC 9995: sign the hash, keep the payload where it belongs

COSE Hash Envelope • Standards Track • became an RFC on 13 Jul 2026

A SELF-DESCRIBING DIGEST

ORIGINAL

SBOM • firmware scan result

HASH SHA-256

COSE_Sign1

payload	digest
258	hash algorithm
259	preimage type
260	payload location

VERIFY without original

DISCOVER optional preimage

TRANSFER less data

The signature authenticates the digest and its protected metadata.

WHAT IT ENABLES

01 Fast signature validation

The original payload is not required to verify the envelope signature.

02 Optional discovery

A verifier can retrieve the preimage and confirm its hash when needed.

03 Cryptographic agility

The protected header identifies both hash and signature choices.

258 REQUIRED

259 / 260 OPTIONAL

WHY SCITT CARES

RFC 9943 §6.2 allows large or sensitive Statements to be signed by hash. SCRAPI -11's POST /entries example uses labels 258–260 for an SPDX SBOM.

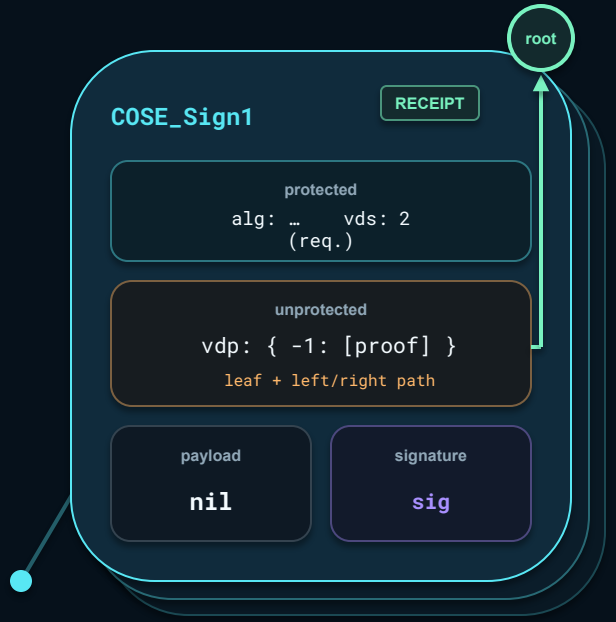
Sources RFC 9995 • publication history • SCRAPI example

CCF Profile and Joint Statements in SCITT - Amaury Chamayou

CCF profile for COSE Receipts

CCF-specific VDS and inclusion proof encoding for SCITT transparency services.

SCITT WG · DRAFT-04 · REQUESTED VDS ASSIGNMENT 2



Henk Birkholz · Antoine Delignat-Lavaud
Cedric Fournet · Amaury Chamayou

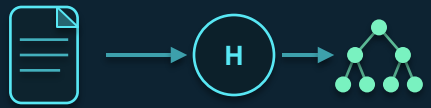
draft-ietf-scitt-receipts-ccf-profile-04 /
24 June 2026

Why CCF needs a distinct VDS profile

CCF leaves carry execution and commit evidence that RFC 9162 leaves do not.

RFC 9162

serialized log entry



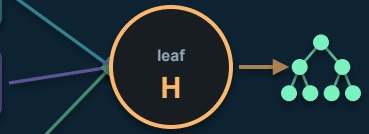
no CCF-specific fields

CCF_LEDGER_SHA256

internal-transaction-hash
complete CCF ledger entry

internal-evidence
revealable commit evidence

data-hash
SCITT Signed Statement



Bind the transaction

internal-transaction-hash commits to the complete CCF ledger entry.

Persist before consensus

internal-evidence supports early persistence; receipts wait for full commit.

CCF leaf construction

The proof carries all three values; only data-hash is SCITT application data.

CCF LEAF

CBOR ARRAY

internal-transaction-hash

32-byte bstr · H(complete CCF ledger entry)

internal-evidence

tstr, 1..1024 bytes · revealable CCF commit evidence

data-hash

32-byte bstr · H(SCITT Signed Statement)

`H(tx_hash // H(evidence) // data_hash)`

internal-transaction-hash

complete CCF ledger entry

application + internal details

H

32-byte digest

internal-evidence

TSTR 1..1024 BYTES

execute

TEE

persist

evidence

consensus

commit

issue

receipt

pre-consensus persistence

receipt after commit

COSE receipt encoding

vds is protected; vdp carries the inclusion proof; the payload is nil.

COSE_Sign1 / SECTION 4 CDDL

PROTECTED

```
1:   alg
     int
395:  vds
      2
```

UNPROTECTED

```
396: vdp
  -1:
  inclusion
    [ proof₀, proof₁, … ]
    proof = { 1: leaf, 2: path }
```

PAYLOAD

~~nil~~

root

detached

SIGNATURE

verify_cose

(receipt, root)

RETURNS BOOL

FLOW

payload = nil root = compute_root(proof) verify_cose(receipt, root)

Inclusion proof verification

Leaf hash, ordered path fold, then COSE verification with the computed root.

1 LEAF HASH

- `tx_hash`
- `H(evidence)`
- `data_hash`

`h = H(tx // H(ev) // data)`

2 PATH FOLD

`for each [left, sibling]`

`left = true  h = H(sibling // h)`

`left = false  h = H(h // sibling)`

repeat to the root

3 COSE VERIFY

`verify_cose(receipt, root)`

RETURNS BOOL

PAYLOAD	INDEX	SIGNATURE INPUT
nil	L/R path bits	computed root

Joint Statement extension proposal

Amaury Chamayou – Microsoft

Yogesh Deshpande - Arm

Motivation

- Artifact signed by multiple parties
  - Same payload
  - Different issuers, subject, algorithms, validity ranges..
- Example: component firmware, signed by:
  - Component manufacturer
  - Device manufacture
  - Enterprise deployment authority
- Each statement can be made transparent individually: RFC9943
- We propose a way to make a **single Transparent Statement, with multiple authorities**

Impact to Standards

- Strict extension
  - Allow COSE_Sign single statement to become transparent
- Unambiguous for single-issuer statements
  - MUST use COSE_Sign1
- Short additional document
- Key points
  - ALL signatures must be verified by the TS
  - Registration policy must pass on EACH (signature + signature-specific header)

Impact to Existing Implementations

- The change is entirely incremental
- RFC9943-compliant implementations remain RFC9943-compliant
- Adding support for multi-signature/joint statement registration
  - Dispatch on tag #6.98
  - For every signature in the statement
    - Apply the same registration procedure as for a COSE_Sign1 in RFC9943
  - There’s no step 3!
- Receipt unchanged
- Note: registration in the VDS captures an order of signatures in the joint statement.

Reviews welcome

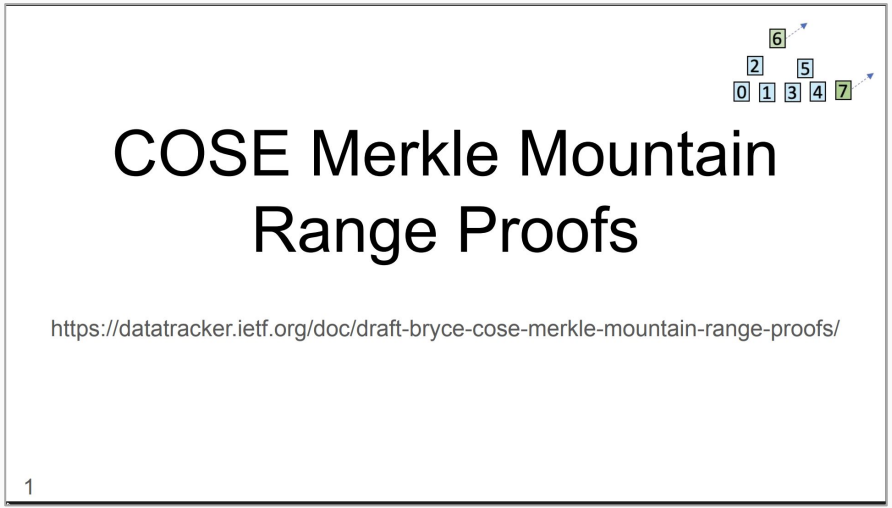
Link to the new draft:
https://achamayou.github.io/draft-chamayou-scitt-cose-sign/draft-chamayou-scitt-cose-sign.html

MMR Profile - Working Group Adoption

Jon Geater

We've been here before

First presented at IETF 121 alongside CCF draft

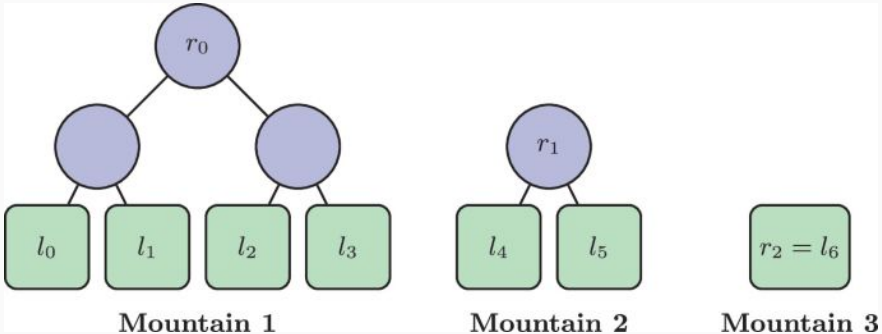


https://datatracker.ietf.org/meeting/121/materials/slides-121-cose-cose-merkle-mountain-range-proofs-intro-01

Why do we need another profile?

The Merkle Mountain Range VDS

- Fast appends
- Co-ordination free writers for HA & uptime
- Easy replication for auditors
- Simple operations for pruning of old data
- No special hardware



“Merkle Mountain Ranges are Optimal” - J Bonneau et al:
https://link.springer.com/chapter/10.1007/978-3-032-01878-6_6

Document status

Good things:

- Functionally complete, aligned to CCF Profile and SCITT.
- BCP furniture etc done.
- Substantially reviewed by related experts.
- Proven open source implementation.

Left to do:

- Question over crypto agility

Datatracker version has been available for a long time:
https://datatracker.ietf.org/doc/draft-bryce-cose-receipts-mmr-profile/

ID-source, -03 ready:
https://github.com/robinbryce/draft-bryce-cose-receipts-mmr-profile/blob/main/draft-bryce-cose-receipts-mmr-profile.md

Hackathon Report

SCITT · WG ADOPTION PROPOSAL

Canonical Payload Binding for Signed Statements

The statement-side twin of the receipt profiles. In charter. No recharter required.

Steven Mih · Action State Group

hackathon report: 2 slides · coordinates in appendix

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

HACKATHON / INTEROP · 1 OF 2

Seven parties ran. Everything public, everything pinned.

Every 'ran' row links a public record — the interop index is the source of truth.

7	4	2	6/6 + 24/24
independent parties	codebases, byte-agreement	receipt profiles, one statement	bidirectional cross-runs

GlyphZero delegation	both directions ran
APS decision records	bidirectional, pinned
Microsoft CCF · two-TS	both receipts verify
PermitReceipt × MachineMandate	pre-registered protocol · results after owners' review
EP receipts	3 independent computations, 1 digest
GAR kernel record	deployed code · chain 3/3 · leaf 166
A2A boundary seal	TCK 100% MUST · close scheduled

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

HACKATHON / INTEROP · 2 OF 2

Every party re-derived the same layer

The join key each landed on — before any pair could agree on bytes.

AAC	SHA-256(JCS(payload − self-ref fields))
GlyphZero	subject_digest 0b4da06b… — 2 JCS impls, same bytes
EP receipts	8cf0c36e… — 3 codebases, same bytes
ORPRG	exact canonical request bytes — joined by typed reference
APS	NFC + code-point sort + JCS + SHA-256
GAR	leaf = SHA-256(BYTES of id), not hex — found the hard way
Two-TS	one digest, two logs, two receipt profiles

THE FINDING Seven parties, different payloads, one missing document: payload → bytes → identifier → Receipt.

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

WHY NOW

The clock is real. The answers are private.

Non-determinism × governance × fragmentation — the gap is the binding layer.

01

EU AI Act Art. 12

record-keeping enforceable Aug 2 — nine days

02

Agents can't be replayed

evidence = byte-committed snapshot at the effect

03

Convergence in vendor toolkits

Microsoft receipts on RFC 8785 JCS · LangChain · AutoGen · CrewAI · Nous · OpenLineage

04

In this room, today

AI MBOMs + CRA compliance — asking to use SCITT

N PRIVATE ANSWERS

Every statement draft restates:
canonicalization · identifier · exclusion set · hex-vs-bytes · leaf · citation.

Each slightly differently.

N × N bilateral debugging.
Auditors: N procedures, or none.

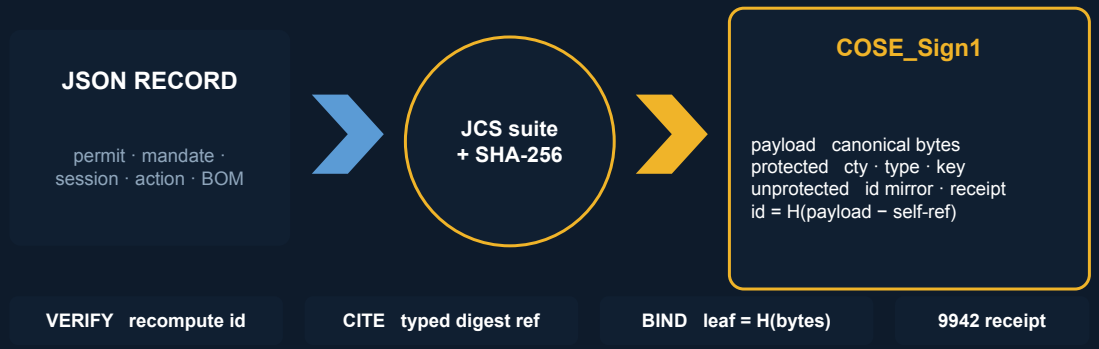
Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

COSE · PAYLOAD BINDING

The same shape you just adopted — one layer over

RFC 9995 binds a hash. This binds structured payload bytes + a derived identifier — and stops there.

A SELF-DERIVING IDENTIFIER



Verifiers MUST recompute; a carried identifier is never trusted.

WHAT IT ENABLES

01 One verification procedure

payload-blind: authentic · unmodified · existed-when · cites-what

02 Records cite records

cross-vendor, zero bilateral agreements

03 Registries, not lists

suites + artifact types · Specification Required

WHY SCITT CARES

Receipts have two profiles here (CCF adopted, MMR proposed). Statement construction has N private answers — one per draft on this list. This makes it one.

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

SCOPE DISCIPLINE

This document never knows what a payload means

OUT	Payload vocabularies
OUT	Verdicts & effects
OUT	Agent semantics
OUT	Application meaning
OUT	Registration policy
OUT	Transports (SCRAPI's)

SEMANTICS land in whichever group is chartered for them — and compose on top by citation.

Not an application profile. The charter's non-goal excludes payload CONTENT formats — this defines none.

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

THE CLEAN DIVISION

One container. Open competition for the slots.

THE CONTAINER — Canonical Payload Binding

IN CHARTER NOW

construction rules + registries · done once · owned by the WG

THE SLOTS — compete on top, keep their own venues, authors, semantics

Agent Action Capsule	mine — one slot · stays individual
Kernel session records	platform
AI MBOMs · CycloneDX	this session
Compliance receipts	Microsoft AGT + frameworks
PermitReceipt · MachineMandate	authorization
Conversation containers	vCon-adjacent
CRA compliance comms	this session
Every future record type	a registry entry, not a new layer

Provenance, plainly: this layer started in my draft. Generalizing it across seven counterparties stripped the agent vocabulary — that's how I knew it was the layer.

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

WHY IN CHARTER

The WG answered this seven months ago

Same class of document. Same process. Same charter.

CHARTER TEXT

“a standard format for authenticity data … independent verification of claims at a (much) later point”

Non-goal excludes payload CONTENT formats (BOMs).

This defines none — it binds whatever an issuer brings.

THIS WG, THIS CYCLE

CCF receipts profile:

CfA 15 Dec 2025 (2-week list call)
→ WG -00 Feb · WGLC May
→ IESG 1 Jul 2026

Semantics-free profile. No recharter. 6.5 months.

THE SIBLING

RFC 9995 — published 12 days ago.

Binds an arbitrary payload by hash. Zero payload semantics.

Routing decided by generality of layer — this one is registration- and receipt-specific: it belongs here.

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

SUMMARY · THE PROPOSAL

Adopt the container now. Semantics go where they’re chartered.

NOW

Adopt the binding profile

Statement construction + registries — payload-neutral, semantics-free, in charter. The statement-side twin of the receipt profiles.

IN CHARTER

NEXT

Semantics → their venues

Agent records (my draft included) → AUDIT when it charters. Conversations → vCon-adjacent work. Each composes by citation — if and when it wants to. Nothing here requires anyone to.

NOT THIS ASK

EITHER WAY

The container is done

Whatever the Future-of-WG discussion decides — more payload slots or not — the construction core is finished, small, and needed by every draft on this list. Happy to contribute in that discussion.

SEPARABLE

One verifier for N record types · records cite records · every player competes for its slot on one layer.

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

THE ASK

Is there interest in adopting a payload-neutral binding profile — semantics explicitly out — as a starting point for WG work?

Confirmed by the standard call on the list.

PROCESS	RFC 7221 · sense of the room today · chairs' 2-week list call · adoption = starting point, change control to the WG
VEHICLE	fresh payload-neutral draft · authorship forming, open to this room · -00 derived from AAC · review pool + implementer base already exist: the seven hackathon parties
STANDING OFFER	if the WG recharters toward more payload slots, I'll help draft the charter text — separate from today's ask

Sources: INTEROP.md · RFC 7221 · SCITT charter · datatracker histories

Thank you

See Appendix for
Hackathon details, coordinates & citations

Incoming New Use Cases

AAC × AEP · APPRAISAL BOUNDARY

Composed appraisal, without trust laundering.

Anton Sokolov · Tyche Institute

record validity ≠ platform validity

The negative branch of the interop readout Steve just showed.

Boundary case: record verifies; platform appraisal fails

Keep the verified fact. Deny the composed claim. Report the failed leg.

APPEND-ONLY RECORD

AAC + SCITT · PASS

COSE ✓ inclusion ✓ binding ✓

SESSION APPRAISAL

AEP + RATS · FAIL

quote ✓ bind ✓ fresh nonce ✕

COMPOSED = record valid ∧ platform valid ∧ digest binds **PASS ∧ FAIL = FAIL**

KEEP verified SCITT record

DENY platform-backed claim

REPORT failed leg + reason

APPEND-ONLY INCLUSION Was this record in the history?

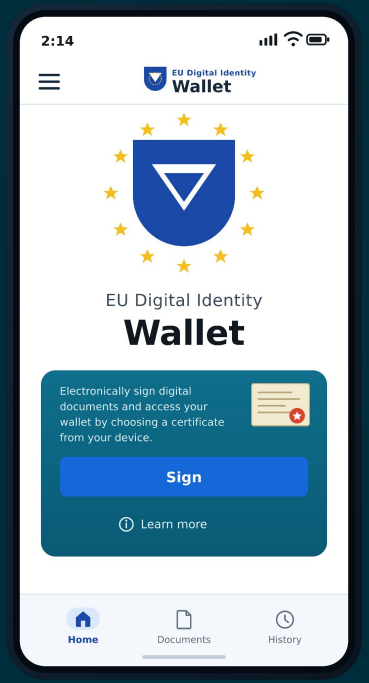
NONCE FRESHNESS Is this evidence fresh for me, now?

The log has a long memory. The nonce has a short one. Green checks don't spread by proximity.

STATUS · positive AAC→AEP cross-run completed · freshness DENY = separate AEP gate vector · vTPM · AAC & AEP: individual I-Ds

Recharter the substrate — not the agent

EUDI Wallet → holder binding → Trusted list → MachineMandate → AgentActionEndorser → **AAC/SCITT** ↔ AEP/RATS



OpenID4VCI · OpenID4VP · SD-JWT VC

MachineMandate is a research profile, not an EUDIW credential type.

SCITT COULD OWN

the interoperable evidence seam

1. Extend beyond software / firmware to accountable digital actions
2. Registration-policy profiles + receipts across VDS profiles
3. CBOR / COSE composition: reuse RFC 9995 + typed digest refs

POSSIBLE BoF / NEW WG

agent delegation and domain semantics

1. Mandate, delegation + endorsement vocabulary
2. CAN / WHO / WHAT / AUDIT accountability policy
3. Domain policy + platform-appraisal semantics

Can SCITT charter the evidence seam — without chartering the agent?

STATUS · phone path running · AAC×AEP cross-verified · four-slot composition draft · vTPM · AAC & AEP: individual I-Ds

CoEvolution

A comprehensive Trustworthy Framework for Connected Machine Learning and Secure Interconnected AI Solutions

Evangelos Haleplidis - haleplidis@athenarc.gr

Website: https://coevolution-project.eu/

Funded by the European Union

This presentation has received funding from the European Union's Horizon Europe research and innovation actions under grant agreement No 101168560. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them

Securing the AI Lifecycle - CoEvolution project

- Security, Trust, and robustness integrated across all AI lifecycle phases:
  - Design
  - Training/Testing
  - Deployment/Inference
- Based on security-, privacy-, and explainability-by-design principles
- Enables trusted, AI models with:
  - Self-monitoring and attack detection
  - Real-time incident reporting to the CoEvolution runtime system

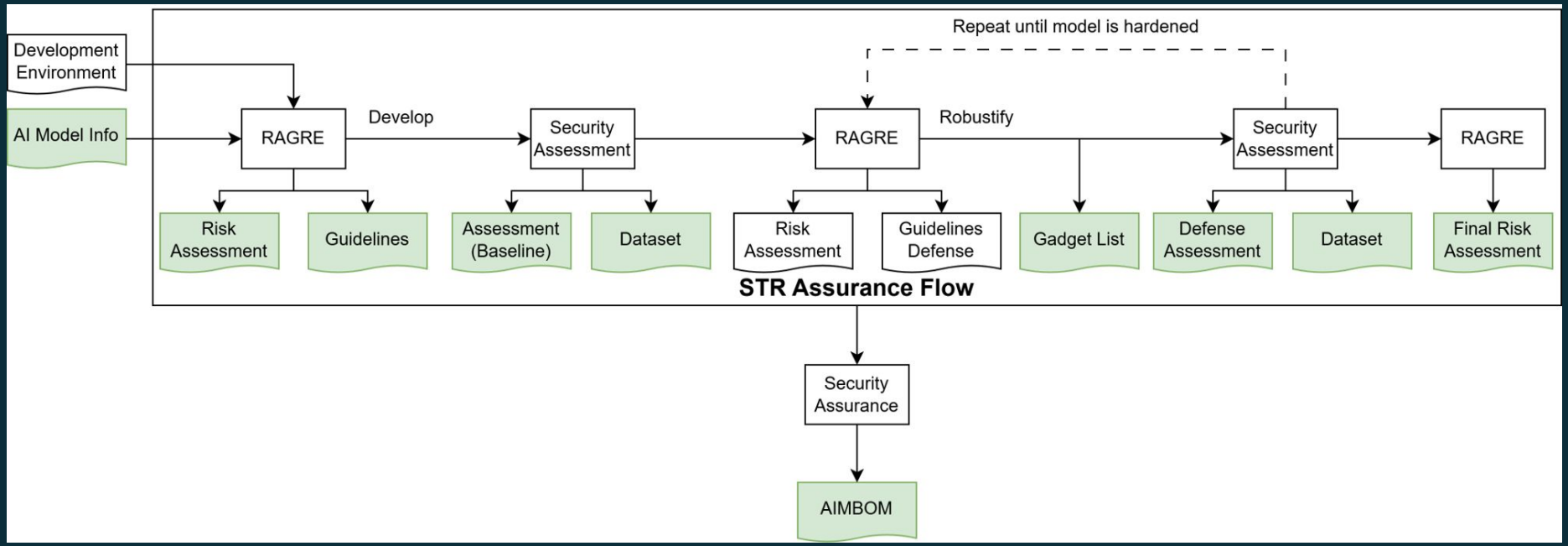
AI Model Bill of Materials (AI MBOM)

- Inspired by Software BOMs enriched with
  - Data provenance
  - Model details
  - Ethical compliance
- Describes datasets, dependencies, owners, and security-related information
- Acts as a carrier of credentials throughout the model lifecycle
- Promotes transparency in robustness, trust, explainability, and fairness

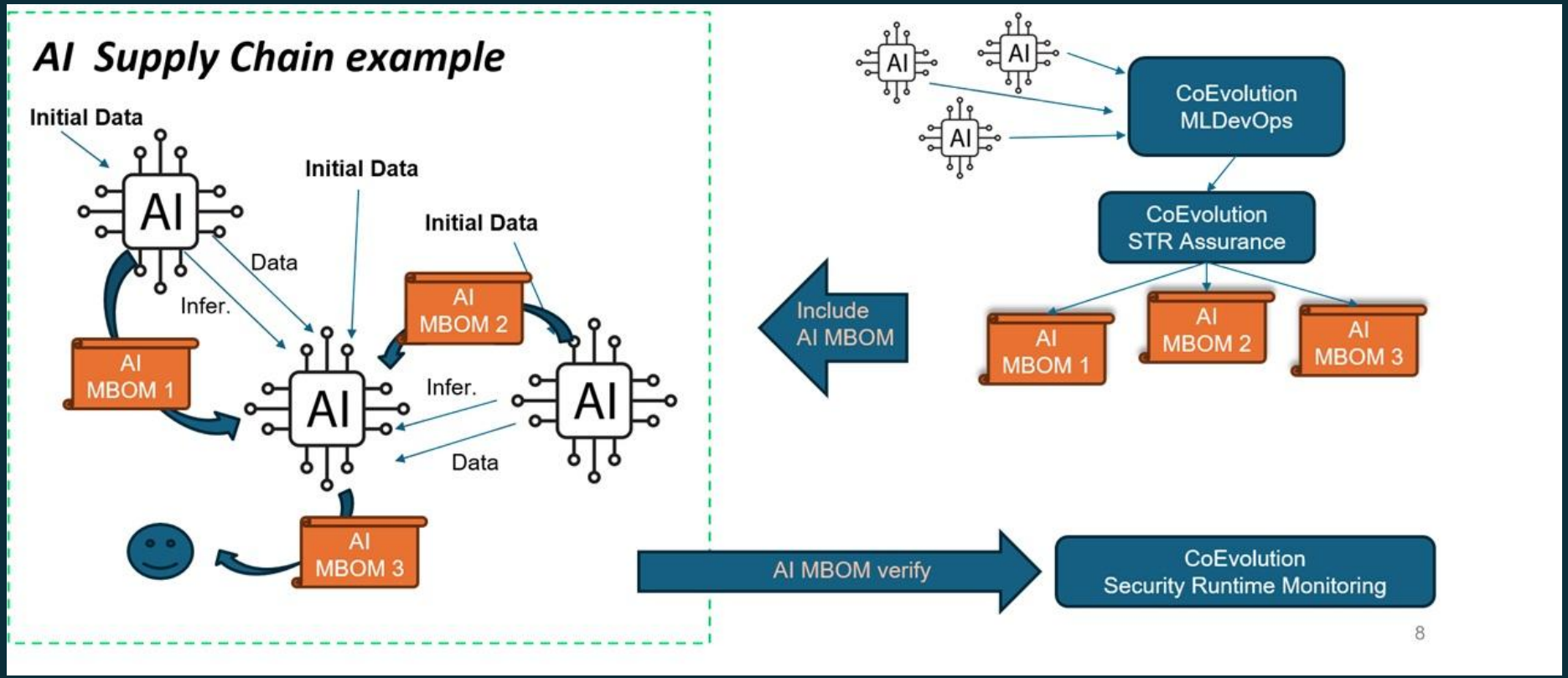
AI Model Bill of Materials (AI MBOM) - Properties

- Has the following properties:
  - Reproducibility
  - Accountability
  - Assessment-Audit
  - Security and Trust
  - Ethical and Responsible AI
  - Uniqueness
  - Risk Assessment

AIMBOM Generation

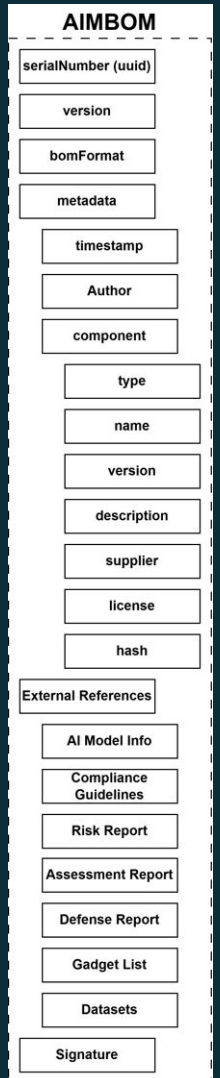


AI Supply Chain Example



AIMBOM Structure

- AIMBOM structure is a set of interconnected **CycloneDX** BOM documents
  - One main AIMBOM document that acts as a central point
  - AI Model Info
  - Compliance Guidelines
  - Risk Report
  - Assessment Report
  - Defense Report
  - Gadget List
  - Datasets



The SCITT angle

- We want to utilize the AIMBOM as a carrier of credentials.
- It can be exchanged with other AI systems
- Can provide trust that the model has undergone a number of assessments and robustification phases and is now secure for such attacks

SCITT

- The AIMBOM is a collection of documents
- Each document is a statement in the eyes of SCITT
- Sign and store these statements using the transparency service

Software Supply Chain Communication
Gregor Bransky

Software Supply Chain Communication - for Compliance

Cyber Resilience Act enters into force to make Europe's cyberspace safer and more secure

PRESS RELEASE

Publication 10 December 2024

Software Supply Chain Communication - for Compliance

Cyber Resilience Act enters into force to make Eu

PRESS R
Publicati

How does the CRA address free and open-source software?

The CRA recognises that, to foster the development and deployment of free and open-source software, special attention should be paid to the nature of the different development models of software distributed and developed under free and open-source software licences.

This is why only **free and open-source software** that is made available on the market, and therefore supplied for distribution or use in the course of a commercial activity, falls in scope of the Cyber Resilience Act. Notably, the provision of products with digital elements qualifying as free and open-

Software Supply Chain Communication - for Compliance

Cyber Resilience Act enters into force to make Eu How does the CRA address free and open-

EXECUTIVE ORDER 14028, IMPROVING THE NATION'S CYBERSECURITY

PRI
Put

Software Supply Chain Security Guidance +

Cybersecurity Labeling for Consumers +

Workshops & Call for

Improving the Nation's Cybersecurity: NIST's Responsibilities Under the May 2021 Executive Order

Overview | Completed Assignments | Latest Updates

supplied for distribution or use in the course of a commercial activity, falls in scope of the Cyber Resilience Act. Notably, the provision of products with digital elements qualifying as free and open-

Software Supply Chain Communication - for Compliance

Software Supply Chain Communication - for Compliance

Cyber Resilience Act enters into force to make

How does the CRA address free and open-

EXECUTIVE ORDER 14028, IMPROVING THE NATION'S CYBERSECURITY

Secure Software Development Framework SSDF

FINAL RULE

U.S Security and Exchange Comission

Cybersecurity Risk Management, Strategy, Governance, and Incident Disclosure

Software Supply Chain Communication - for Compliance

Cyber Resilience Act enters into force to make
Eu How does the CRA address free and open-

EXECUTIVE ORDER 14028, IMPROVING THE NATION'S CYBERSECURITY

Secure Software Development Framework SSDF

FINAL RULE

U.S Security and Exchange Comission

Safeguarding artifact integrity across any software supply chain

Software Supply Chain Communication - for Compliance

Cyber Resilience Act enters into force to make

EU How does the CRA address free and open-

EXECUTIVE ORDER 14028, IMPROVING THE NATION'S CYBERSECURITY

Secure Software Development Framework SSDF

FINAL RULE

U.S Security and Exchange Comission

Chainguard

citi

CLOUD NATIVE COMPUTING FOUNDATION

DATADOG

Google

intel

KUSARI

THE LINUX FOUNDATION

Software Supply Chain Communication - for Compliance

Cyber Resilience Act enters into force to make
Eu How does the CRA address free and open-

EXECUTIVE ORDER 14028, IMPROVING THE NATION'S CYBERSECURITY

Secure Software Development Framework SSDF

FINAL RULE

U.S Security and Exchange Comission

Chainguard citi CLOUD NATIVE COMPUTING FOUNDATION DATADOG

National Telecommunications and Information Administration
U.S. Department of Commerce

Section 515 Standards:
Guidelines for Ensuring and Maximizing the Quality, Objectivity, Utility, and Integrity of Information Disseminated by the National Telecommunications and Information Administration

Software Supply Chain Communication - Why SCITT

1. publish supply chain data, decentralized, globally interoperable
2. Identify compliance critical communication (c3) within the ecosystem

Software Supply Chain Communication - The Ecosystem

Software Supply Chain Communication - The Ecosystem

enisa EUROPEAN UNION AGENCY FOR CYBERSECURITY

Nationaal Cyber Security Centrum
Ministerie van Veiligheid en Justitie

Rijksinspectie Digitale Infrastructuur
Ministerie van Economische Zaken en Klima

Port of Rotterdam

1. **Market Surveillance Authority**

NixOS

Software Supply Chain Communication - The Ecosystem

1. **Market Surveillance Authority**

2. **FOSS - Up & Downstream**

Software Supply Chain Communication - The Ecosystem

enisa EUROPEAN UNION AGENCY FOR CYBERSECURITY

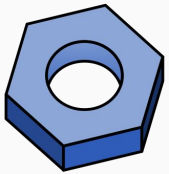
Nationaal Cyber Security Centrum
Ministerie van Veiligheid en Justitie

Rijksinspectie Digitale Infrastructuur
Ministerie van Economische Zaken en Klima

Port of Rotterdam

1. **Market Surveillance Authority**





NixOS

SkyPilot

Determinate Systems

2. **FOSS - Up & Downstream**

3. **Dependents aka Manufacturers**

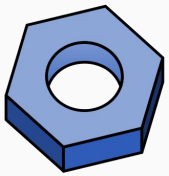
Software Supply Chain Communication - The Ecosystem

Nationaal Cyber Security Centrum
Ministerie van Veiligheid en Justitie

Rijksinspectie Digitale Infrastructuur
Ministerie van Economische Zaken en Klima

Port of Rotterdam

1. **Market Surveillance Authority**



2. **FOSS - Up & Downstream**

3. **Dependents aka Manufacturers**

4. **Your Forges**

Software Supply Chain Communication - The Ecosystem

enisa EUROPEAN UNION AGENCY FOR CYBERSECURITY

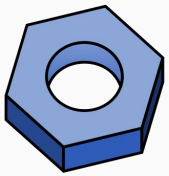
Nationaal Cyber Security Centrum
Ministerie van Veiligheid en Justitie

Rijksinspectie Digitale Infrastructuur
Ministerie van Economische Zaken en Klima

Port of Rotterdam

1. Market Surveillance Authority





2. FOSS - Up & Downstream

3. Dependents aka Manufacturers



4. Your Forges



5. White AI Blackhats

Software Supply Chain Communication - 4 ideas

1. publish supply chain data, decentralized, globally interoperable
2. Identify compliance critical communication (c3) within the ecosystem
    a. by Market Surveillance Authorities (MSAs)
    b. by FOSS Projects Up- and Downstream
    c. by proprietary dependents

Software Supply Chain Communication - 1. idea

1. publish supply chain data, decentralized, globally interoperable

2. FOSS - Up & Downstream

3. Dependents aka Manufacturers

4. Your Forges

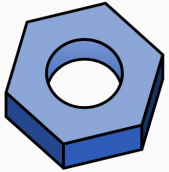
Software Supply Chain Communication - 1. idea

1. publish supply chain data, decentralized, globally interoperable

Standards Badges - SB

Allow FOSS projects to flag on their repositories which standards they implement





2. FOSS - Up & Downstream

3. Dependents aka Manufacturers



4. Your Forges

Software Supply Chain Communication - The Ecosystem

enisa EUROPEAN UNION AGENCY FOR CYBERSECURITY

Nationaal Cyber Security Centrum
Ministerie van Veiligheid en Justitie

Rijksinspectie Digitale Infrastructuur
Ministerie van Economische Zaken en Klima

Port of Rotterdam

1. **Market Surveillance Authority**

NixOS

2. Identify compliance critical communication (c3)
   a. with Market Surveillance Authorities (MSAs)

Software Supply Chain Communication - The Ecosystem

enisa EUROPEAN UNION AGENCY FOR CYBERSECURITY

Nationaal Cyber Security Centrum
Ministerie van Veiligheid en Justitie

Rijksinspectie Digitale Infrastructuur
Ministerie van Economische Zaken en Klima

Port of Rotterdam

1. **Market Surveillance Authority**

NixOS

2. Identify compliance critical communication (c3)
   a. with Market Surveillance Authorities

MSAs should have a PKI

Software Supply Chain Communication - The Ecosystem

2. FOSS - Up & Downstream

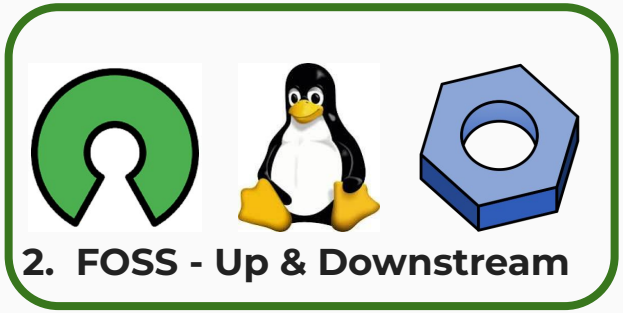
4. Your Forges

2. Identify compliance critical communication (c3)
   b. by FOSS Projects Up- and Downstream

Software Supply Chain Communication - The Ecosystem

I) Write message

I)

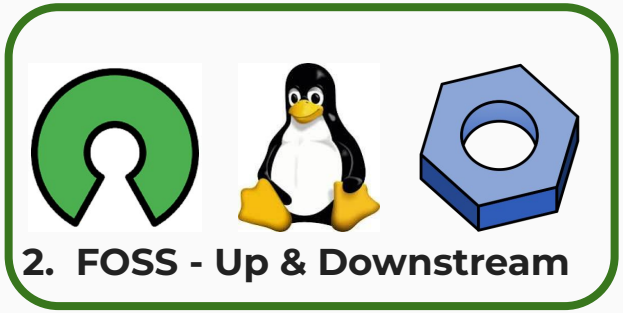


NixOS

2. Identify compliance critical communication (c3)
   b. by FOSS Projects Up- and Downstream

Software Supply Chain Communication - The Ecosystem

I) II)



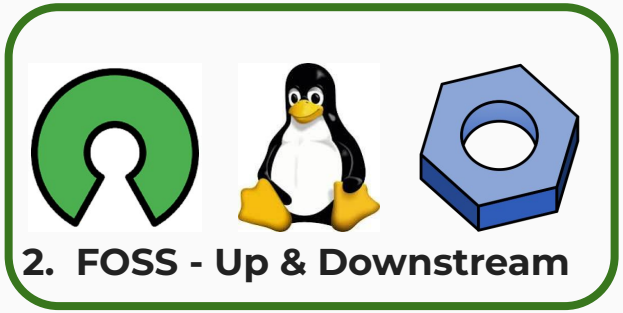
I) Write message
II) Publish hash of message with forge account

NixOS

2. Identify compliance critical communication (c3)
b. by FOSS Projects Up- and Downstream

Software Supply Chain Communication - The Ecosystem

I) II) III)



NixOS

I) Write message
II) Publish hash of message with forge account
III) Send mail to security@ with pointer to hash

2. Identify compliance critical communication (c3)
   b. by FOSS Projects Up- and Downstream

Software Supply Chain Communication - The Ecosystem

I) II) III)

IV)

I) Write message
II) Publish hash of message with forge account
III) Send mail to security@ with pointer to hash
IV) Run centrality measure on forge account of sender

2. Identify compliance critical communication (c3)
   b. by FOSS Projects Up- and Downstream

Software Supply Chain Communication - The Ecosystem

I) Use Package

I)

3. Dependents aka Manufacturers

4. Your Forges

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Software Supply Chain Communication - The Ecosystem

I) Use Package
II) get reset of properties of package (reset has UID)

I) II)

3. Dependents aka Manufacturers

4. Your Forges

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Software Supply Chain Communication - The Ecosystem

I) Use Package
II) get reset of properties of package (reset has UID)
III) send donation with UID

I) II) III)

3. Dependents aka Manufacturers

4. Your Forges

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Software Supply Chain Communication - The Ecosystem

I) Use Package
II) get reset of properties of package (reset has UID)
III) send donation with UID
IV) send c3 with list of UIDs

I) II) III) IV)

3. Dependents aka Manufacturers

4. Your Forges

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Software Supply Chain Communication - The Ecosystem

I) Use Package
II) get reset of properties of package (reset has UID)
III) send donation with UID
IV) send c3 with list of UIDs
V) Asses validity by UIDs

V)

NixOS

I) II) III) IV)

SkyPilot

Determinate Systems

3. Dependents aka Manufacturers

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Codeberg

4. Your Forges

Software Supply Chain Communication - The Ecosystem

I) Use Package
II) get reset of properties of package (reset has UID)
III) send donation with UID
IV) send c3 with list of UIDs
V) Assess validity by UIDs

V)

NixOS

I) II) III) IV)

SkyPilot

Determinate Systems

3. Dependents aka Manufacturers

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Codeberg

4. Your Forges

Software Supply Chain Communication - The Ecosystem

I) Use Package
II) get reset of properties of package (reset has UID)
III) send donation with UID
IV) send c3 with list of UIDs
V) Asses validity by UIDs

V)

NixOS

I) II) III) IV)

SkyPilot

Determinate Systems

3. Dependents aka Manufacturers

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Codeberg

4. Your Forges

Software Supply Chain Communication - The Ecosystem

I) Use Package
II) get reset of properties of package (reset has UID)
III) send donation with UID
IV) send c3 with list of UIDs
V) Asses validity by UIDs

V)

NixOS

I) II) III) IV)

SkyPilot

Determinate Systems

3. Dependents aka Manufacturers

2. Identify compliance critical communication (c3)
   c. by proprietary dependents

Codeberg

4. Your Forges

In scope - technical specs

1. Security and privacy guardrails for all stakeholders
2. Technical Soundness
3. Applicability

Out of scope - Interference with governance

1. Interference with the governance of FOSS Projects, thus:
    a. Free choice of regulation to be adhered
    b. Free choice of revealed information
    c. Free choice of payment channels
    d. Free choice of centrality measures
    e. dataformats for payload content
    f. etc. etc.
2. Create obstacles for
    a. business innovation
    b. market creation

Open Questions - Scope

1. Which Software supply chain regulation should be considered?
   a. that Is already out there?
   b. that Is in the making?
2. Are there best practices?
3. Which RFC are to be considered? (e.g. secure channels RFC 9781)
4. Whom to talk to?
5. What did we miss?

Open Questions - Caveats

1. Trust is a social question, technology can help, not answer it.
2. Malicious fixes - downstream targets
3. Fraud
4. Impersonation and relay attacks
5. Open Source Software Stewards - a vehicle for money laundering?
6. What did we miss?

Open Questions - Caveats

1. Malicious fixes - downstream targets
2. Fraud
3. Impersonation and relay attacks
4. Open Source Software Stewards - a vehicle for money laundering?
5. What did we miss?

Asks

1. Are you familiar with:
    a. EO 14028
    b. SSDF
    c. SEC Cyber Rule
    d. SLSA
    e. NTIA
    f. other regulation ..
    or compliance tooling?

Asks

1. Are you familiar with:
    a. EO 14028
    b. SSDF
    c. SEC Cyber Rule
    d. SLSA
    e. NTIA
    f. other regulation ..
       or compliance tooling?
2. How to put badges on repositories?

Asks

1. Are you familiar with:
    a. EO 14028
    b. SSDF
    c. SEC Cyber Rule
    d. SLSA
    e. NTIA
    f. other regulation ..
    or compliance tooling?
2. How to put badges on repositories?
3. Wanna come talk to ORCs - Code and Compliance 27 Oct Brussels?

Asks

1. Are you familiar with:
   a. EO 14028
   b. SSDF
   c. SEC Cyber Rule
   d. SLSA
   e. NTIA
   f. other regulation ..
      or compliance tooling?
2. How to put badges on repositories?
3. Wanna come talk to ORCs Code and Compliance 27 Oct Brussels?
4. How to identify AI hats?

Working Group Future

Future of the working group

TL;DR: Are we almost done?
SCITT Architecture and SCRAPI are complete, VDS docs are advanced.

However: We now have a flurry of interest in new streams of work

So, options:

1. Finish the current work in flight and assume an orderly wind-down
2. Re-charter based on the new documents and inputs
3. Establish informative registry of use cases/payload formats that allow other groups to _use_ SCITT but do not expand the WG officially.

Propose to cut everything below here. No time, not WG focused.

APPENDIX

Appendix — artifacts & coordinates

Everything in this report re-runs from these. A name is not evidence; a digest is — every claim links a public record.

REPOS & INDEX

Interop index (all tracks, statuses, links):
github.com/action-state-group/agent-action-capsule/blob/main/INTEROP.md

Reference implementation + spec sources:
github.com/action-state-group/agent-action-capsule (Apache-2.0/BSD-3)

Substrate verifier (payload-agnostic): github.com/action-state-group/scitt-cose

Transparency service (reference): github.com/action-state-group/capsule-anchor

Producer + adapters: github.com/action-state-group/capsule-emit

LIVE SERVICES & VECTORS

Live log: anchor.agentactioncapsule.org

Verifier endpoint: verify.actionstate.ai

Conformance vectors (immutable): scitt-cose tag vectors-ietf126 (32 Class-1, named failing stages)

Two-TS artifacts: scitt-cose interop/ccf/shared-vector.json (v2)

A2A boundary tuple: capsule-emit tag a2a-boundary-ietf126

PUBLIC INTEROP RECORDS

Two-TS example (both receipts): ietf-wg-scitt/examples PR #4 (approved by CCF profile maintainer)

pyscitt vector pinning: microsoft/scitt-ccf-ledger PR #424

Delegation cross-runs (both directions):
karthik-titech/cross-org-delegation-registry issue #3

APS bidirectional cross-run: mirjak/audit-bof-preparation issue #9

A2A bilateral close: action-state-group/capsule-emit issue #29

Platform-attestation bundle (pinned): tyche-institute/aep-pcr16-vector release v1.0-prior-vector

THE DRAFT FAMILY (individual I-Ds)

Record profile: datatracker.ietf.org/doc/draft-mih-scitt-agent-action-capsule (-02)

Selective disclosure companion:
…/draft-mih-scitt-agent-action-capsule-sel-disc

Bilateral attestation: …/draft-mih-agent-bilateral-attestation (-01)

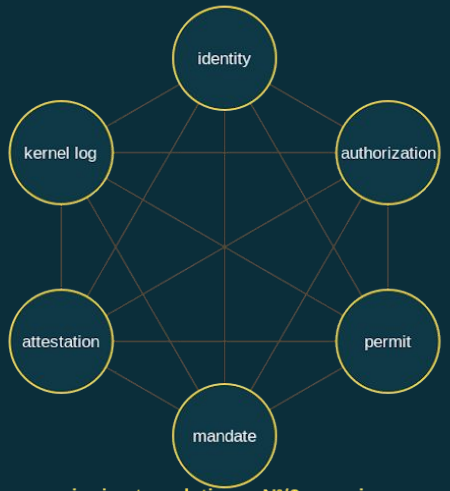
Accountability composition (4 authors):
…/draft-mih-sato-agent-accountability-composition

Regulated-evidence mapping (QuadX): FCA Obligations Mapping v1.0 — by request / results page

Individual Internet-Drafts: not WG documents, not RFCs. Substrate: SCITT architecture RFC 9943 · COSE Receipts RFC 9942.

IETF 126 Hackathon: 1 claim, many implementations

- The claim: SCITT is the substrate for verifiable agent-action records — no architecture changes needed. Agent evidence is a payload class this WG's specs already carry.
- An Agent Action Capsule: a signed, content-addressed record of what an agent actually did — anchored via SCITT, verifiable by a party who trusts neither the agent nor its operator.
- Format of this report: what ran (graded honestly) → what broke (findings for the profile authors) → what it signals for the WG.
- Everything shown re-runs from public kits.



pairwise translation: ~N²/2 mappings

new systems arrive faster than old ones die

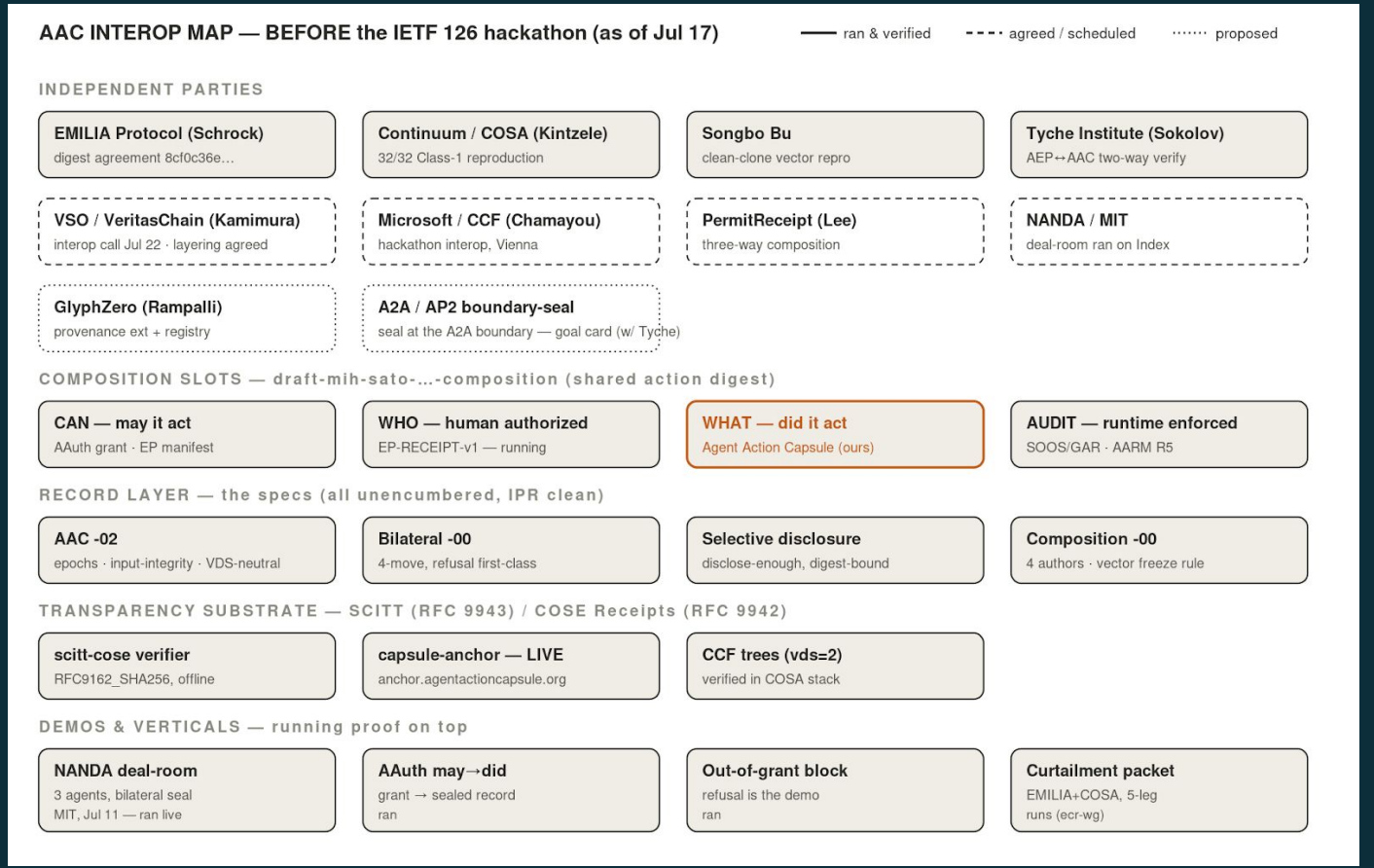
VS

one common record: N mappings — every pair composes

what ran this week: five record models, one digest, zero pairwise adapters

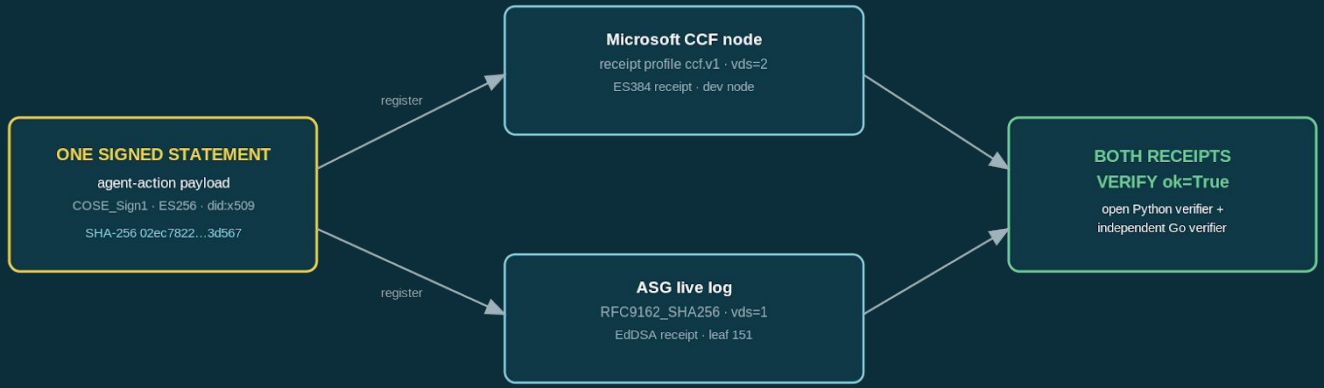
IETF 126 Hackathon Interop

IETF 126 Hackathon project: one claim, tested across independent implementations



Headline: one statement, two Transparency Services

with Amaury Chamayou (Microsoft)



same envelope, two VDS profiles in conjunction — receipt profiles compose, as well as stand alone

- ONE COSE_Sign1 Signed Statement (ES256, did:x509 — Microsoft-signed) registered on TWO independent Transparency Services: a CCF node (ccf.v1, vds 2) and our live RFC 9162 log (vds 1, anchor.agentactioncapsule.org).
- BOTH COSE receipts verify — each under independent verifiers (scitt-cose Python + clean-room Go; pyscitt's side one command away via PR #424, whose test inputs are our published vectors, byte-identical 4/4).
- Second in-the-wild instance of the MMR-example pattern (MST + DataTrails, ietf-wg-scitt/examples): receipt profiles used in conjunction as well as in isolation. Our transparent statement with both receipts → ietf-wg-scitt/examples PR #4 — APPROVED by the CCF profile maintainer.
- Specs exercised: RFC 9943 architecture · RFC 9942 receipts · SCRAPI register/resolve · CCF vds 2 · RFC9162_SHA256 vds 1. (Payloads inline — RFC 9995 hash-envelope not exercised; a natural next vector.)

Three-way accountability: permit × mandate × capsule

PermitReceipt (Y.B. Lee) × MachineMandate (A. Sokolov) × AAC — a pre-registered evidence protocol

- One protected action (a payment); two authorities commit in DIFFERENT canonical forms by construction — the capsule is the only place both bind, via typed digest references. Digest match alone is NEVER authorization success: binding gates + owner appraisals, all four required.
- The protocol IS the story: frozen input manifest (v0.5, SHA-pinned), owner confirmations on the record, a named independent external replay of the frozen tuple (20/20 · 17/17 · 19/19 on an outside machine), negative cases frozen to a single mutation with a named first-rejecting gate.
- Platform weld: the capsule's outcome digest folds into a TPM PCR-16 quote — the chip attests the same digest the record carries (AEP × RATS).
- Execution runs under the owners' complete pre-run freeze record (pre-registered protocol, frozen manifests, immutable coordinates); results are stated after owner review — in the owners' words, not ours.

Composition across independent parties

Graded under each registry's own evidence ladder — claims never outran artifacts

- Cross-registry agent contracts (MIT NANDA × AI Catalog): agents from two discovery worlds negotiated; sealed bilateral records cite both identities by digest, anchored — GET /verify returns true for anyone.
- Platform (Tyche AEP × RATS): TPM quote binding our capsule re-verified on our machine, stock tpm2-tools + an independent parser — the public record's third-party-execution row moved 0 → 1 evidenced. Includes a published finding-withdrawal after byte-level re-check: the discipline works in both directions.
- Record-model interop (APS, draft-pidlisnyi-aps): bidirectional vector cross-runs on the public AUDIT-prep record — they ran vectors-ietf126 (6/6, negatives at declared stages, byte-identical re-runs); we ran their v0.1.0 corpus (24/24, scope boundaries stated, not counted as passes).

- Delegation (Glyph Zero Labs), BOTH directions: they re-ran our provenance vectors (splice negative fails at provenance_ref_binds) and recomputed our subject digest with SURADAR's own RFC 8785 JCS — byte-for-byte; we re-ran their PEDIGREE vectors (over-scope rejected as scope_creep). One shared digest, two complementary layers — independent_interop.
    - QuadX (UK regulatory consultancy) took the AAC × PEDIGREE run artifacts above — the shared subject digest, the splice and scope_creep negatives, the public record and mapped 11 FCA obligations onto them, graded by what a supervisor could verify without trusting the firm. The delegation evidence became their strongest row (SYSC 8, agent-as-delegate). Next slide →

Regulated-evidence mapping — delivered by QuadX AI (22 July)

Domain-expert requirements traceability: 11 FCA/UK obligations → evidence artifacts, graded by verification depth — adjacent to the interop results, not among them

WHAT IT IS

BNPL / consumer-credit AI-agent scenario: 11 obligations (CONC · SYSC · PRIN · DISP · s.21 FSMA · UK GDPR Art. 22) mapped to specific capsule fields and executed artifacts — clause references verified against the FCA Handbook as at 22 Jul 2026; through QuadX's internal regulatory + technical review.

Every row graded: OA operator-asserted · PV participant-verifiable · TPV third-party-verifiable (a stranger — auditor, FCA, court — verifies from artifact + public anchor alone).

Rules as written, not as enforced. Not legal advice; not a conformance determination — the honesty is the method.

WHAT THE GRADING FINDS

Existence, sequence, authorisation, delivery, refusal, and policy-version: third-party-verifiable TODAY with artifacts already exchanged. Quality judgments (reasonableness, fairness, good outcomes) stay supervisory — no cryptography changes that.

SYSC 8 (their addition): a third-party AI agent performing regulated functions IS outsourcing — and the mandate binding (effect.authorization, provenance_ref_binds) is how a firm demonstrates retained control. Strongest evidence in the table: AAC × PEDIGREE, both directions, public record.

DISP 8-week rule: the ABSENCE of a resolution capsule is itself evidence — a missing record more probative than a present one.

WHY IT MATTERS FOR THIS ROOM

A regulated-sector domain expert independently mapped statutory obligations onto SCITT-anchored records and graded what a supervisor could verify without trusting the firm. No UK regime mandates evidence-side verifiability yet — but Consumer Duty outcomes monitoring, SUP 16 reporting, and s.166 skilled-person reviews all pull toward it, and the mapping shows TPV verifiability is available today, ahead of the requirement. Ladder: expert mapping (delivered) → independent auditor review citing row artifacts by digest → POC deployment.

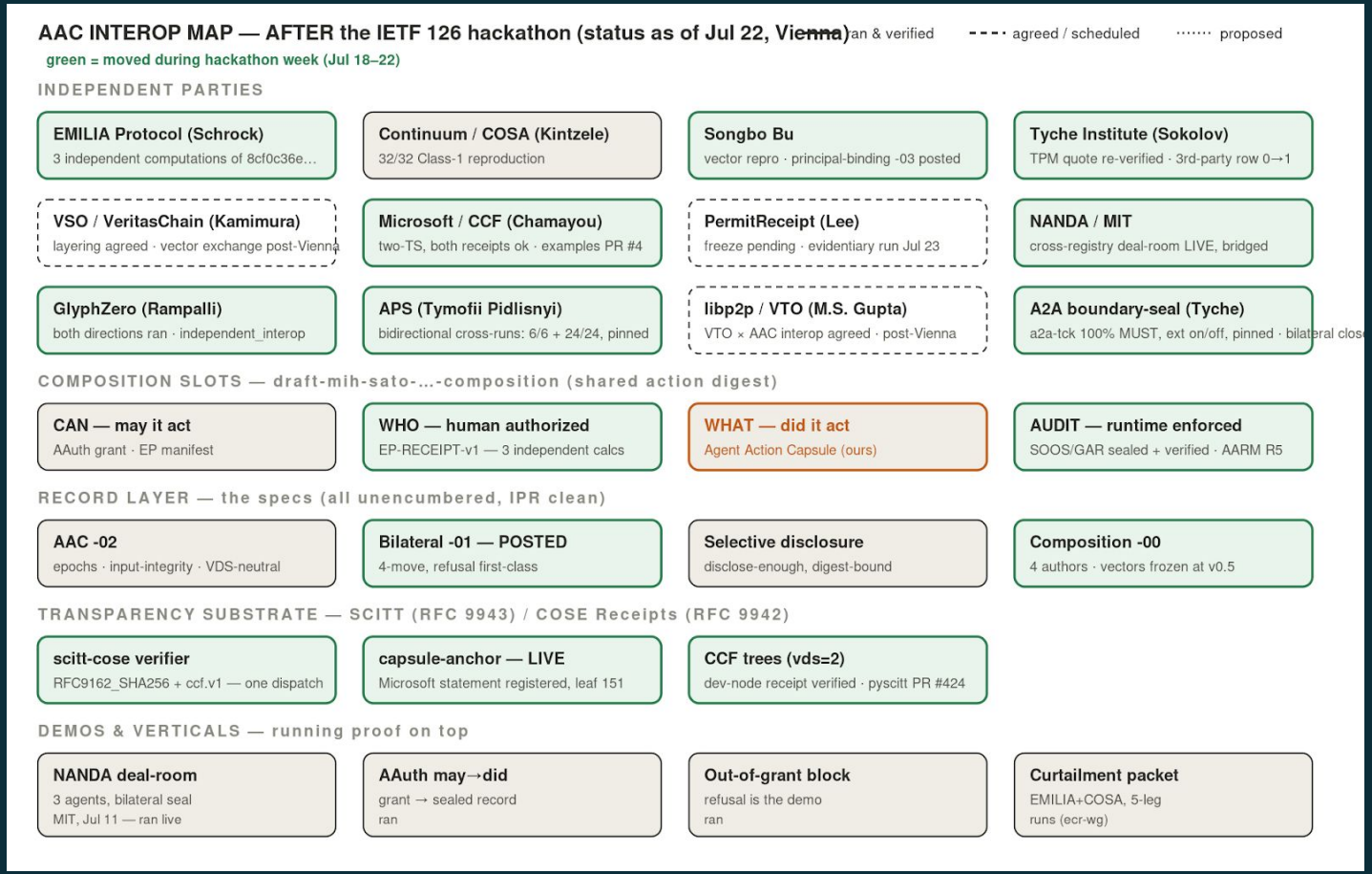
QuadX AI · FCA Obligations Mapping Final v1.0 · authored + internally reviewed by QuadX regulatory and technical leads · evidence rows cite public executed artifacts (issue #3, digests)

More seams exercised

- SOOS / GAR (T. Sato): kernel-governance records → SCITT Signed Statements, sealed + verified end-to-end; canonicalization finding below came from this seam. A REAL Session Block from his deployed code (Apache-2.0, pinned) — chain verified 3/3 against his published implementation, then sealed + anchored (leaf 166)
- EP authorization receipts (I. Schrock): named-human approval cited by digest — the WHO leg; three independent computations of one subject digest (8cf0c36e…) from separately written codebases.
- Security-principal binding (S. Bu, -03 posted this week): verifier-boundary discipline — byte-exact digest-composition rules aligned with what ran here.
- Producer adapters: MCP · Google ADK · agentgateway · LangChain · CrewAI · Goose · Hermes · Dapr ~30 lines each to emit capsules from a framework.

The Readout

IETF 126 Hackathon project: one claim, tested across independent implementations



github.com/action-state-group/agent-action-capsule/blob/main/INTEROP.md

What broke — findings for the profile authors

The credibility core: the specs mostly worked; here is where the edges are

- Dangling pins, twice in 24 hours: a regenerated evidence bundle under an unchanged filename; a force-pushed commit under a pinned SHA. Both caught only by re-hashing; both fixed with immutable coordinates. A name is not evidence; a digest is.
- vds dispatch: read vds from the PROTECTED header only; reject unknown values. Now mirrored in microsoft/scitt-ccf-ledger PR #424.
- Canonicalization: sorted-stringify vs JCS-with-normalization silently diverge (empty members). Stating the normalization rule got four independent codebases to byte-agreement.
- Registration policy is where TS interop actually gates: the CCF node's did:x509 policy could not accept our raw Ed25519 statement — hence the two-TS result runs on THEIR statement. Registration-policy profiles deserve WG attention.
- Issuer binding: bare kid + test issuer ⇒ relying parties cannot authenticate issuers without pre-provisioned mappings. Direction: did:web resolution or x5chain, specified in registration policy.