20th, March 2025

Supply Chain Integrity, Transparency, and Trust (SCITT)

This session is being recorded

Note Well

This is a reminder of IETF policies in effect on various topics such as patents or code of conduct. It is only meant to point you in the right direction. Exceptions may apply. The IETF's patent policy and the definition of an IETF "contribution" and "participation" are set forth in BCP 79; please read it carefully.

As a reminder:

- By participating in the IETF, you agree to follow IETF processes and policies.
- If you are aware that any IETF contribution is covered by patents or patent applications that are owned or controlled by you or your sponsor, you must disclose that fact, or not participate in the discussion.
- As a participant in or attendee to any IETF activity you acknowledge that written, audio, video, and photographic records of meetings may be made public.
- Personal information that you provide to IETF will be handled in accordance with the IETF Privacy Statement.
- As a participant or attendee, you agree to work respectfully with other participants; please contact the ombudsteam (https://www.ietf.org/contact/ombudsteam/) if you have questions or concerns about this.

Definitive information is in the documents listed below and other IETF BCPs. For advice, please talk to WG chairs or ADs:

- BCP 9 (Internet Standards Process)
- BCP 25 (Working Group processes)
- BCP 25 (Anti-Harassment Procedures)
- BCP 54 (Code of Conduct)
- BCP 78 (Copyright)
- BCP 79 (Patents, Participation)
- https://www.ietf.org/privacy-policy/(Privacy Policy)

Note Really Well

- IETF meetings, virtual meetings, and mailing lists are intended for professional collaboration and networking, as defined in the IETF Guidelines for Conduct (RFC 7154), the IETF Anti-Harassment Policy, and the IETF Anti-Harassment Procedures (RFC 7776). If you have any concerns about observed behavior, please talk to the Ombudsteam, who are available if you need confidentiality to raise concerns confident about harassment or other conduct in the IETF.
- The IETF strives to create and maintain an environment in which people of many different backgrounds and identities are treated with dignity, decency, and respect. Those who participate in the IETF are expected to behave according to professional standards and demonstrate appropriate workplace behavior.
- IETF participants must not engage in harassment while at IETF meetings, virtual meetings, social events, or on mailing lists. Harassment is unwelcome hostile or intimidating behavior—in particular, speech or behavior that is aggressive or intimidates.
- If you believe you have been harassed, notice that someone else is being harassed, or have any other concerns, you are encouraged to raise your concern in confidence with one of the Ombudspersons.

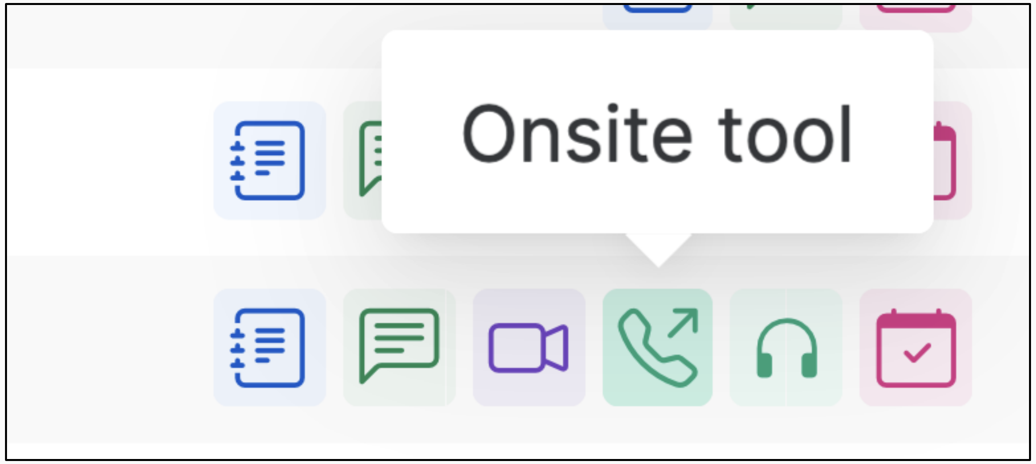
IETF 120 Meeting Tips

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 mic queue
  - participate in shows of 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.



Note taker?

Please assist in taking notes

HedgeDoc

Agenda

● Welcome and Introduction (5 mins):	Chairs
● SCITT Overview (10 mins):	Henk
● Recap since 121 (10 mins):	Steve
● Introduction to Transparency 201 (20 mins)	Jon
● SCRAPI (15 mins):	Jon
● Hackathon Report (10 mins):	Jon
● Next Steps (5 mins):	Chairs
● AOB Open Mic (10 mins):	All
● Wrap-up and Conclusion (5 mins):	Chairs

SCITT Overview

Henk Birkholz <henk.birkholz@sit.fraunhofer.de>

What Is SCITT

- **Working Group Charter**
  The Supply Chain Integrity, Transparency, and Trust (SCITT) WG will define a set of interoperable building blocks that will allow implementers to build integrity and accountability into software supply chain systems to help assure trustworthy operation.

Internet Drafts

- The Architecture draft has reached stability and looking for WGLC
- Focus shifting SCR**API**, validating the architecture is complete

[draft-ietf-scitt-architecture](#)

Ready for WGLC

Receipt

[draft-ietf-cose-merkle-tree-proofs](#)

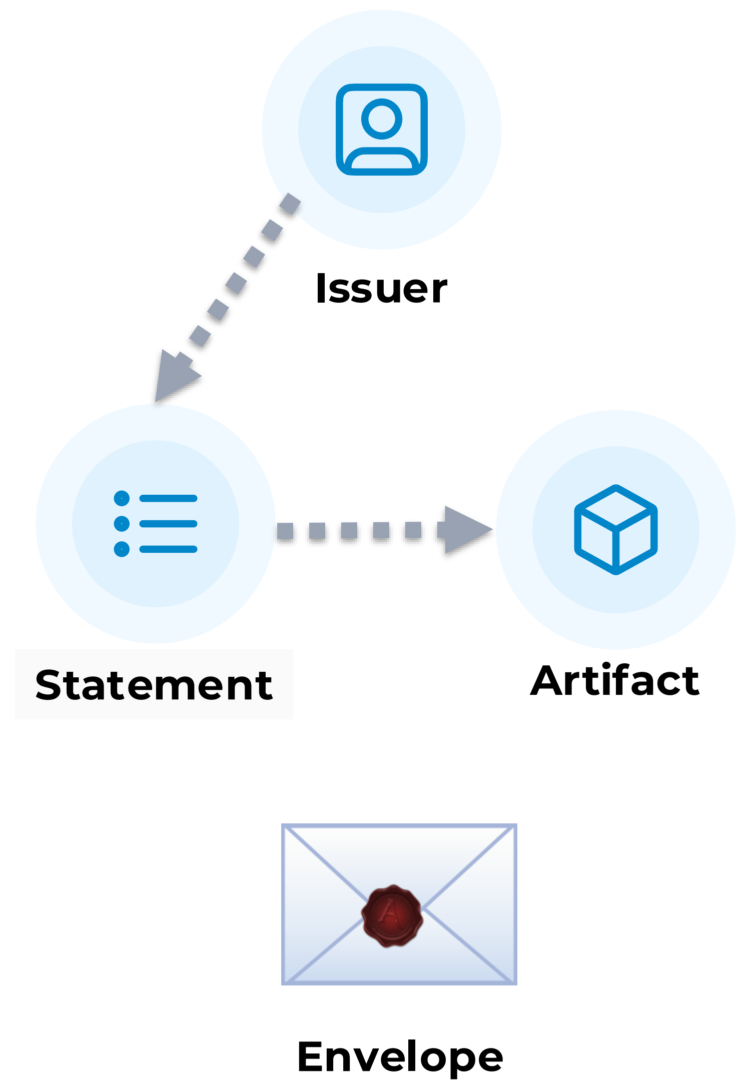
[draft-birkholz-**cose**-receipts-ccf-profile](#)

[draft-bryce-**cose**-merkle-mountain-range-proofs](#)

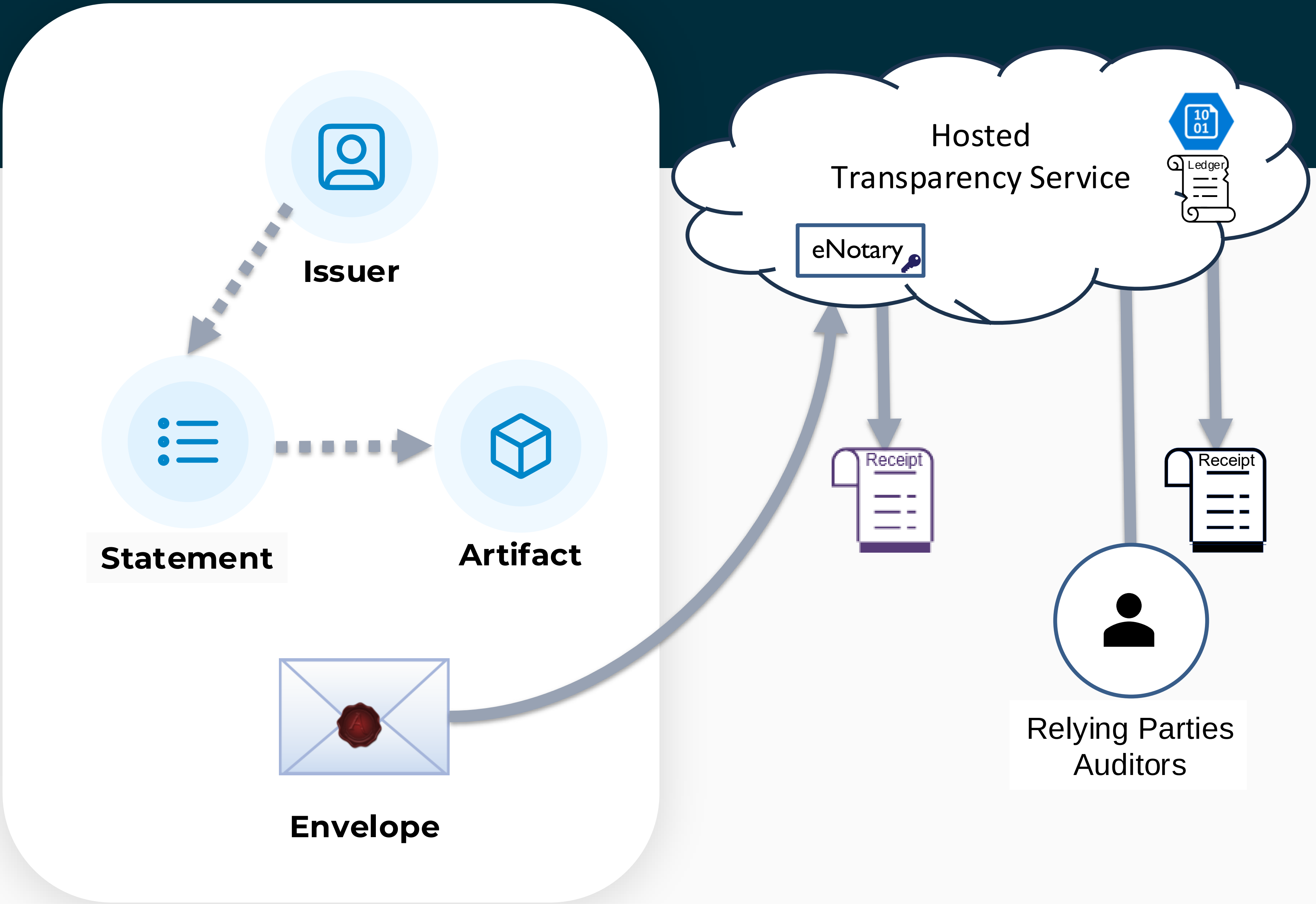
[draft-ietf-**cose**-hash-envelope](#)

[draft-ietf-**scitt**-scrapi](#)

Priority



Issuers make Statements about Artifacts



Signed Statements Registered on a SCITT Transparency Service

```
Signed_Statement =
       #6.18(COSE_Sign1)
```

Envelope

Issuer

Statement

Artifact

Protected_Header

```
&(CWT_Claims : 15) => CWT_Claims
? &(alg      : 1) => int
? &(content_type: 3) => tstr / int
? &(kid      : 4) => bstr
? &(x5t      : 34) => COSE_CertHash
* int => any
```

CWT_Claims

```
issuer  : tstr,
subject : tstr,
*       : int => any
```

```
Signed_Statement =
       #6.18(COSE_Sign1)
```

Envelope

Issuer

Statement

Artifact

Envelope : COSE_SIGN1

```
protected    : bstr .cbor Protected_Header,
unprotected  : Unprotected_Header,
payload      : bstr / nil,
signature    : bstr
```

Protected_Header

```
&(CWT_Claims : 15) => CWT_Claims
? &(alg        : 1) => int
? &(content_type: 3) => tstr / int
? &(kid        : 4) => bstr
? &(x5t        : 34) => COSE_CertHash
* int => any
```

CWT_Claims

```
issuer   : tstr,
subject  : tstr,
*        : int => any
```

```
Signed_Statement =
        #6.18(COSE_Sign1)
```

Envelope

Issuer

Statement

Artifact

Envelope : COSE_SIGN1

```
protected    : bstr .cbor Protected_Header,
unprotected  : Unprotected_Header,
payload      : bstr / nil,
signature    : bstr
```

Protected_Header

```
&(CWT_Claims : 15) => CWT_Claims
? &(alg        : 1) => int
? &(content_type: 3) => tstr / uint
? &(kid        : 4) => bstr
? &(x5t        : 34) => COSE_CertHash
* int => any
```

CWT_Claims

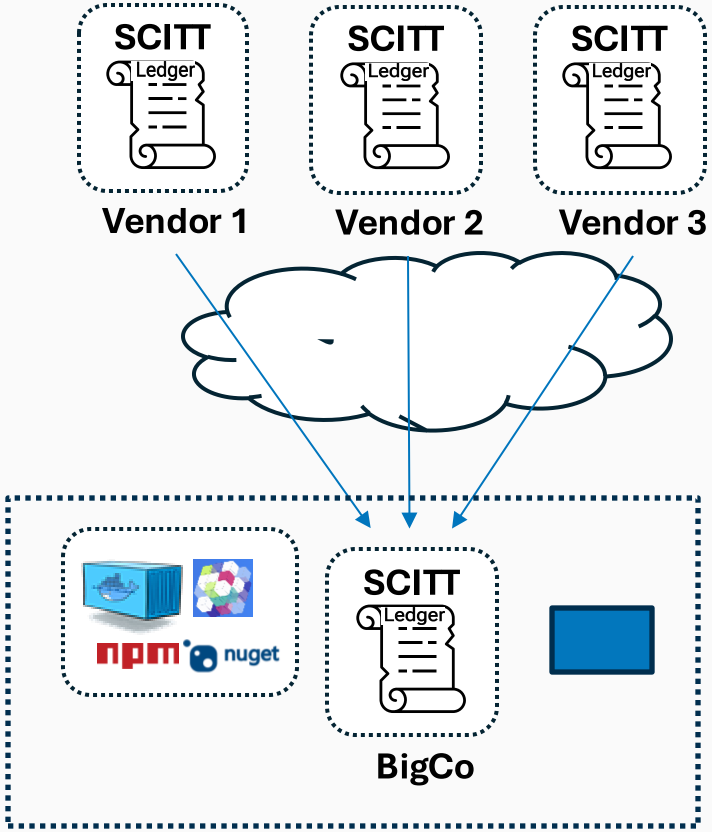
```
issuer  : tstr,
subject : tstr,
*       : int => any
```

Unprotected_Header

```
? &(receipts : 394)  => [+ Receipt]
? &(x5chain  : 33) => COSE_X509
* int => any
```

Trusted Update Streams

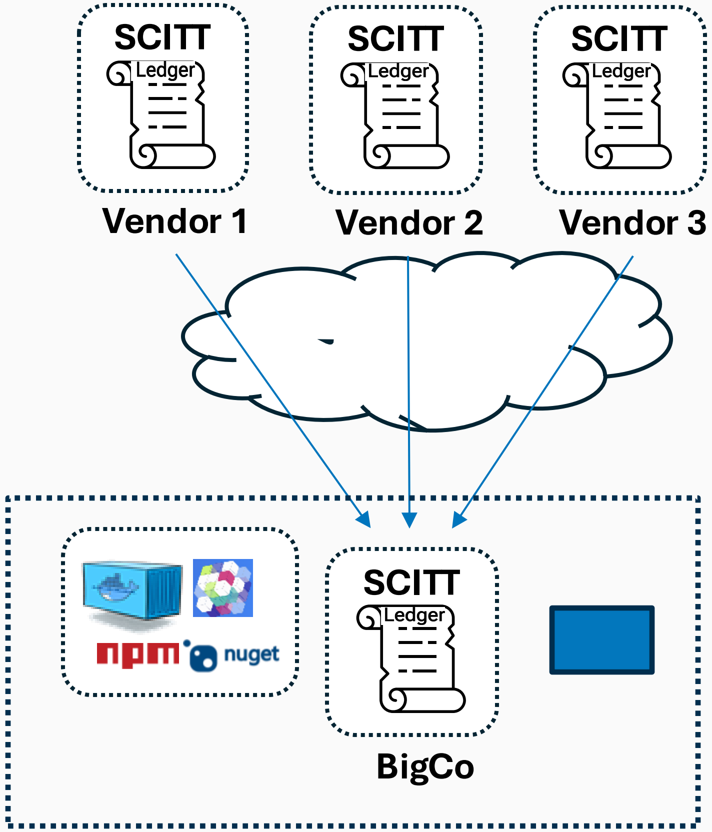
- What is the most current information, about an artifact?
  - Does it have test case data?
  - What certifications does it have?
  - What's the latest version, and is it applicable to "my" environment?
  - What version has "the business" decided is appropriate for its environments?
  - Who is making the statements, and are they trusted for "my" environment?
  - ***"Latest" does not equal the most reliable or safe!***



Trusted Update Streams

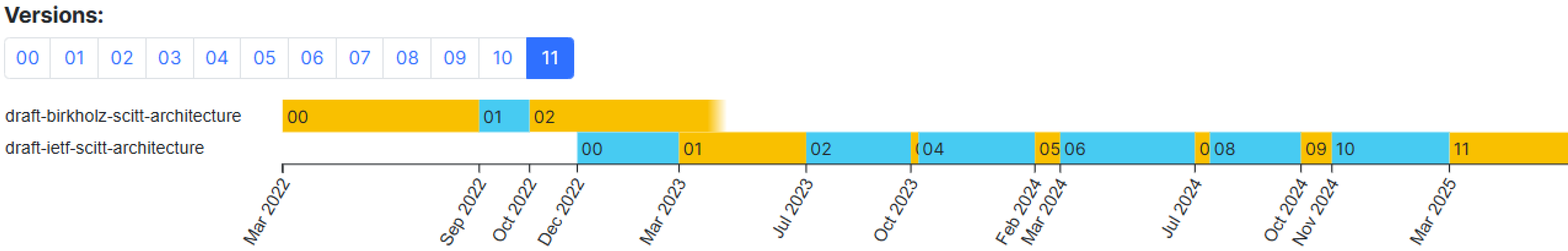
- What is the most current information, about an artifact?
  - Does it have test case data?
  - What certifications does it have?
  - What's the latest version, and is it applicable to "my" environment?
  - What version has "the business" decided is appropriate for its environments?
  - Who is making the statements, and are they trusted for "my" environment?
  - ***"Latest" does not equal the most reliable or safe!***

- SCITT doesn't define content types (but uses them)
- It does enable **Trust** decisions for **Who** published **What** & **When**



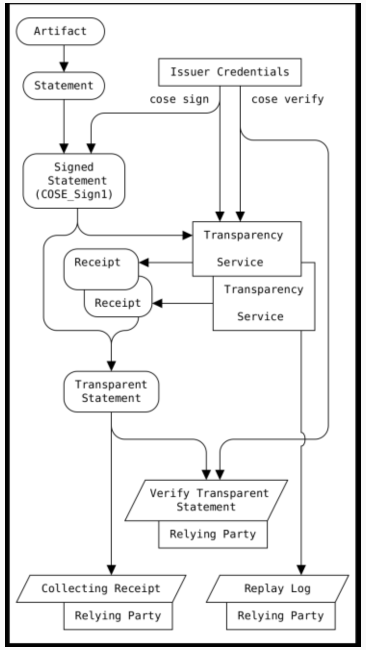
Recap Since 121 Architecture

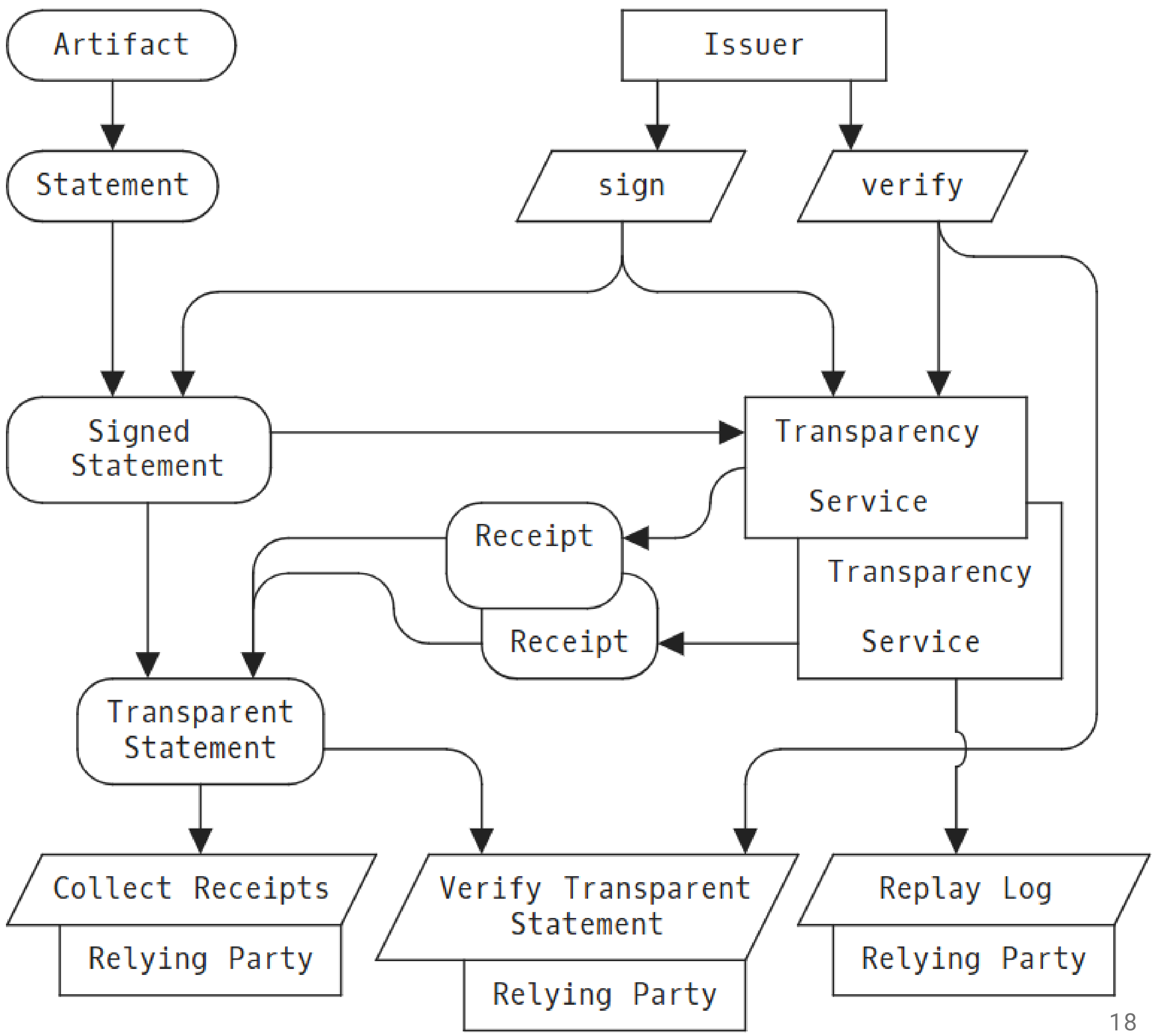
Steve Lasker <steve.lasker@datatrails.ai>



Architectural Overview

- Figure 1 has stabilized – again ☺
  - A picture is a worth a thousands words





Recap Since 121: Architecture

- Revisit reference of Append-Only for VDS #348
- Clarify that the unprotected header MUST be empty before a Signed Statement is included in the Append-only log #349
- Update PII handling advice in security section #346
- Clarify trust anchor wording #345
- Remove URNs #343
- Compress Architecture Diagram #339
- Start refactoring x.509 profile #336
- ...add three eclectic use cases #316

Recap Since 121
SCRAPI

Recap Since 121: SCRAPI

- ...Receipt Exchange Endpoint #70
- Update Resolve Receipt Example #69
- Remove Nonce endpoint #66
- Remove URNs #62
- ...Simplify asynchronous submission flow #60
- Allow 3xx, for redirection, see other etc #58
- Drop 'Issue Statement' endpoint and related considerations #53
- Make authentication stance consistent in the document #52
- Refactor all the RegisterSignedStatement workflows #51
- First tranche of CBOR All The Things #38

Other Drafts...

- draft-ietf-COSE Receipts-08
  - <henk update> Registry being established
  - First two profiles queued:
    - COSE Receipts with CCF-03
    - Merkle Mountain Range for Immediately Verifiable and Replicable Commitments-02
- draft-ietf-cose-hash-envelope-03- covered in COSE session
  - Pre-allocation of COSE Headers

Introduction to Transparency 201

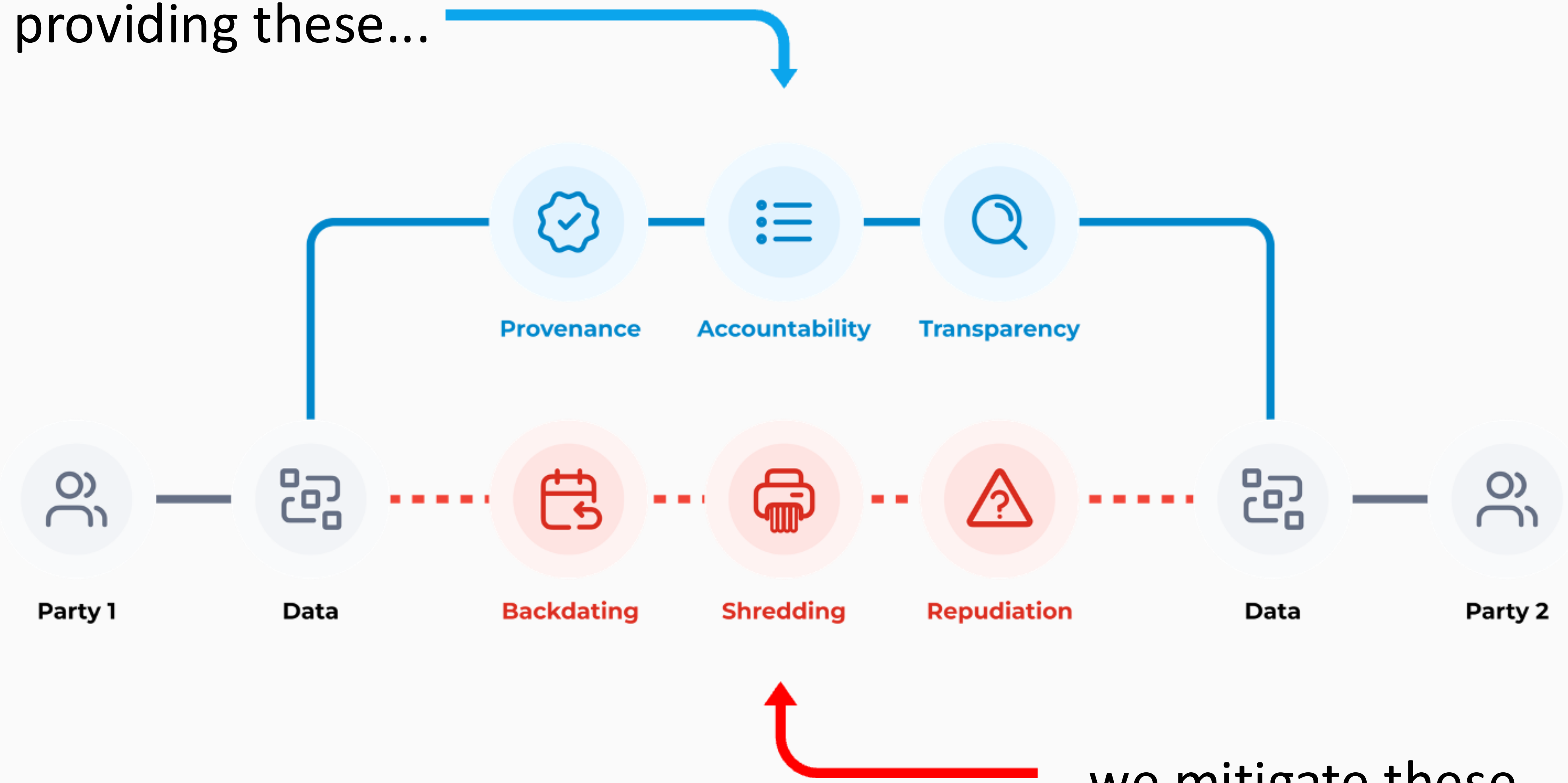
Jon Geater <jon.geater@datatrails.ai>

A Naïve View of Transparency

A Naïve View of Transparency

Paper problems moving to the digital realm

By providing these...



...we mitigate these

Framing

✅ Transparency IS about:

- Verifiable record-keeping (audit trails, cryptographic proofs)
- Ensuring integrity & accountability (traceability without revealing sensitive data)
- Minimizing reliance on trusted third parties (trust through math, not institutions)
- And these things over prolonged times where a lot can change

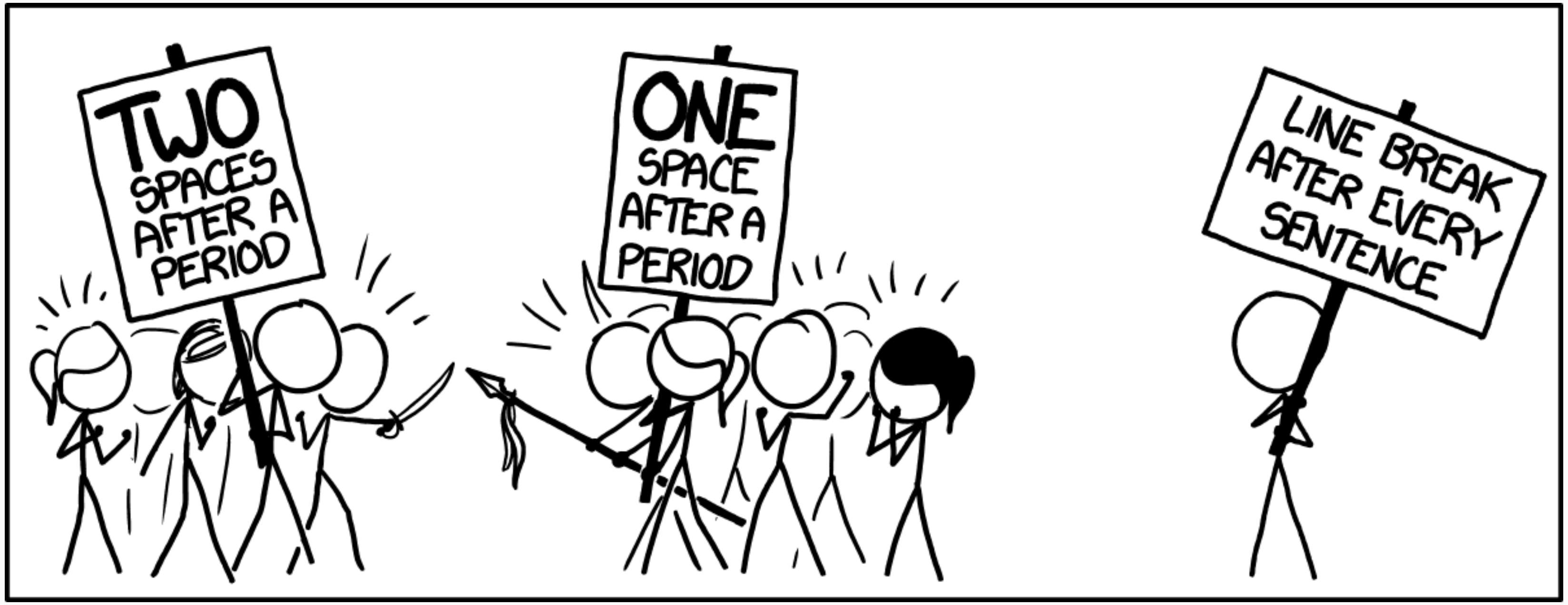
❌ Transparency IS NOT about:

- Unrestricted data sharing (it ensures access control & selective disclosure)
- Data sharing AT ALL, in many cases
- Only centralized registries (it can be decentralized or federated)
- A replacement for privacy (it works alongside confidentiality)
- A replacement for other integrity measures (it augments them)

Just Saying Things Isn't (useful) Transparency

Just publishing things is insufficient:

- If you see a statement, that doesn’t mean anything on its own: it’s just an opinion
- Or more usefully we might want to consider it an ‘attestation’ - a witness statement
    - ...so then you need to know *who* the witness is
    - ...and so ad infinitum



Transparency through verifiability

What we get

- Transparency technologies = Verifiability
- Key properties:
    - Integrity: Once data is recorded, it can't be tampered with.
    - Traceability: A verifiable chain of actions or approvals.
    - Minimal disclosure: Only authorized parties can see what they need.

How we do it

- Tamper-evident logs (Merkle trees, transparency services)
    - Independent of the actual artifact
- Cryptographic receipts (proofs of registration, proofs of inclusion)
    - Independent of the actual artifact
- Trustworthy selective disclosure
    - *Because* the proofs are independent of the actual artifact
- Decentralized transparency (ledger-based registries, DLT where applicable)

Scenarios

Versatile tamper-proofing
Beyond simple sigantures: operates across boundaries and time frames.

Pre-commitment
Prove commitment to some fact which can be held accountable later WITHOUT the need to expose the fact.

Anyone with a copy of the original fact can then verify it later on, *even if the original Issuer doesn't want them to!*

Non-equivocation
Pre-commitment is great, but it goes further: if the specific claim is strongly and uniquely identified

Building blocks

Building blocks enable us to compose systems that use the best of what is offered and provide the set of features they need.

Not everyone needs paranoid privacy.

Not everyone wants the TS and adjacent services to ever see the data.

Some people want to craft a complete transparent registry for files.

SCITT and SCRAPI define the essential core for all of these use cases.

SCRAPI and SCITT Transparency Services

Steve Lasker <steve.lasker@datatrails.ai>

IETF

What Is Stored on the Append-only Log / Ledger

- What started as a simple request:
  - [Clarify what is stored on the ledger #329]()
- Became
  - [Replace Append-only with Verifiable Data Structure #351]()
  - Which opened up:

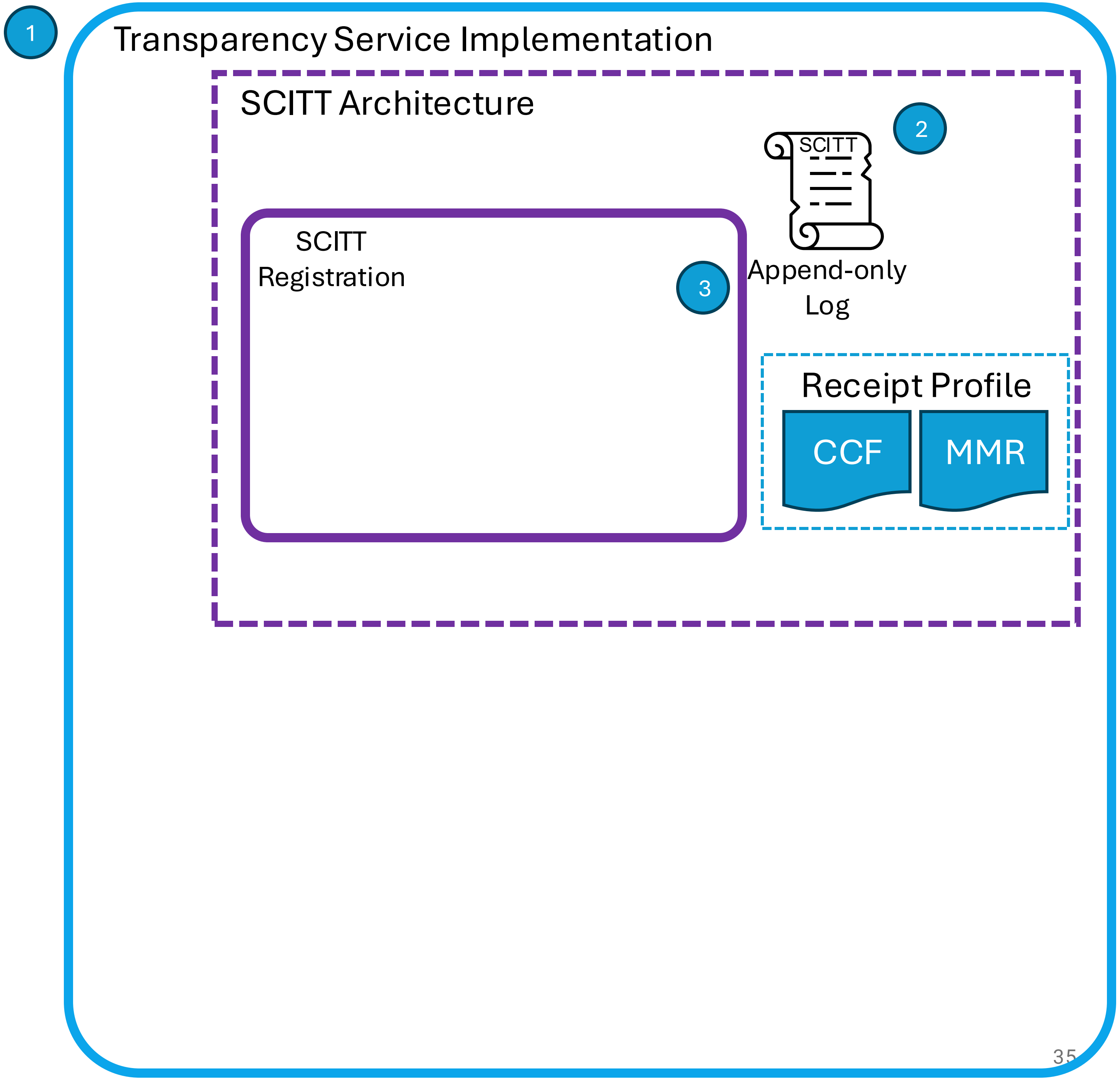
What Is Stored on the Append-only Log / Ledger

1. The Signed Statement?
   - Does that include the `payload`, which can be Bits to Terabytes?
   - What if the `payload` contains PII information?
   - The `payload` *could* be a hash of the pre-image content
   - Does that include the `unprotected header`?
   - What if the service expects or sets properties of the unprotected header?
2. A hash of the signed statement?
   - Is the signed statement saved in “ancillary services”

3. ***“Yes” (1 & 2):*** depending on the Transparency Service Profile Implementation?
   - [COSE Receipts]provides implementers to choose their format
     - [COSE Receipts with CCF]
     - [Merkle Mountain Range for Immediately Verifiable and Replicable Commitments]

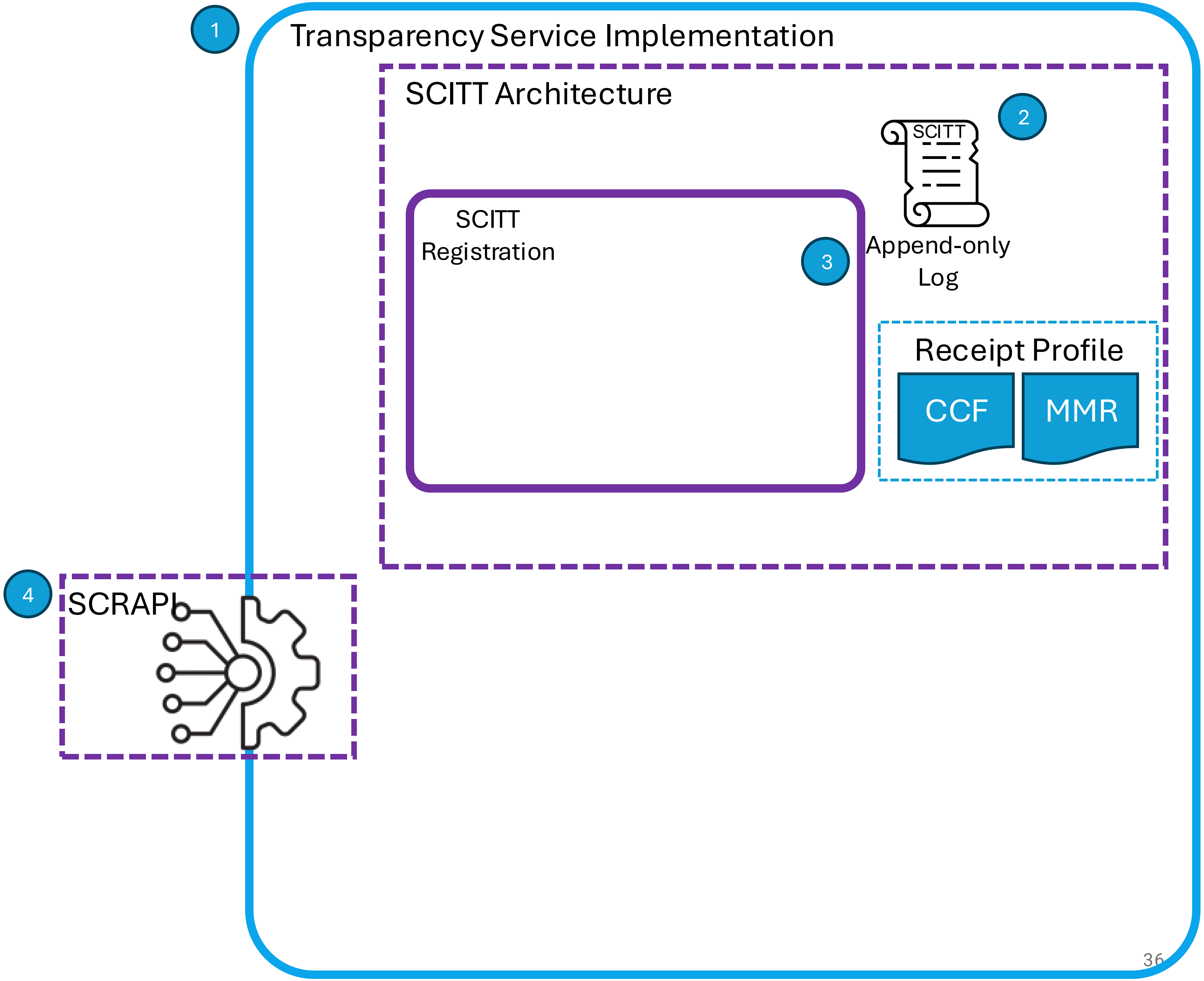
Layering

- Append-only Log
  - Receipt Profile



Layering

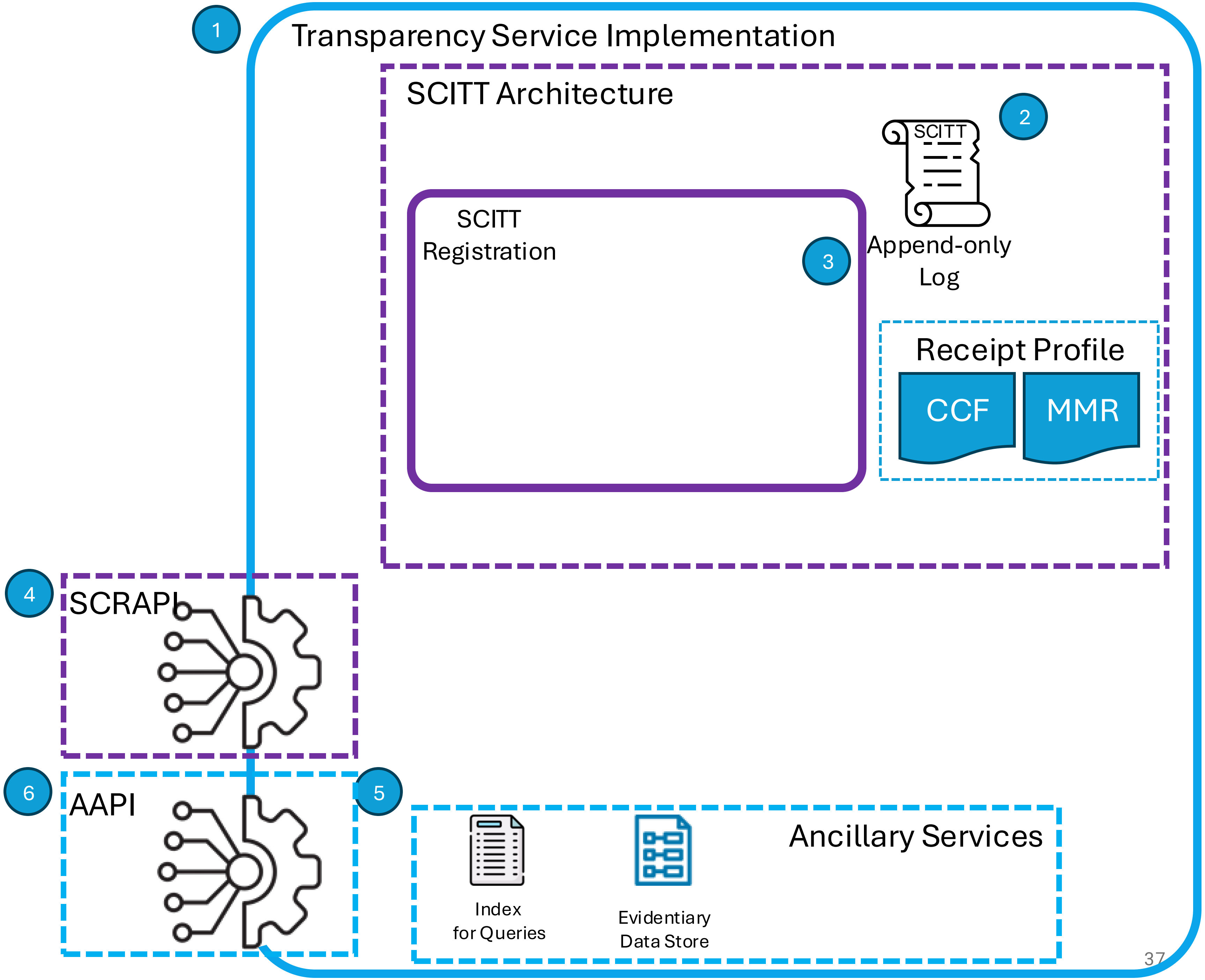
- SCRAPI
- Append-only Log
  - Receipt Profile



Layering

- SCRAPI
- Append-only Log
  - Receipt Profile
- Ancillary Services

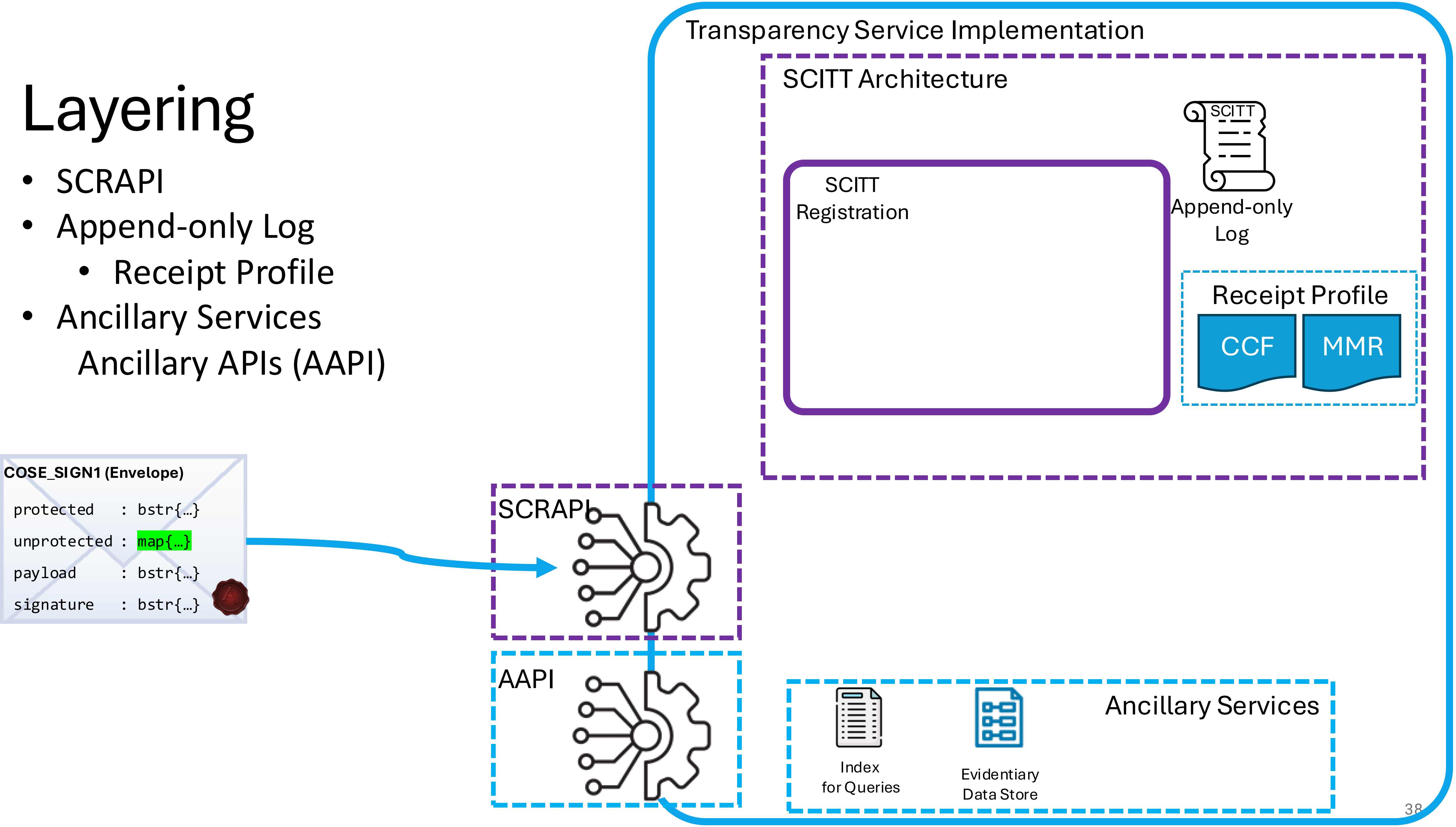
  Ancillary APIs (AAPI)



Layering

- SCRAPI
- Append-only Log
  - Receipt Profile
- Ancillary Services

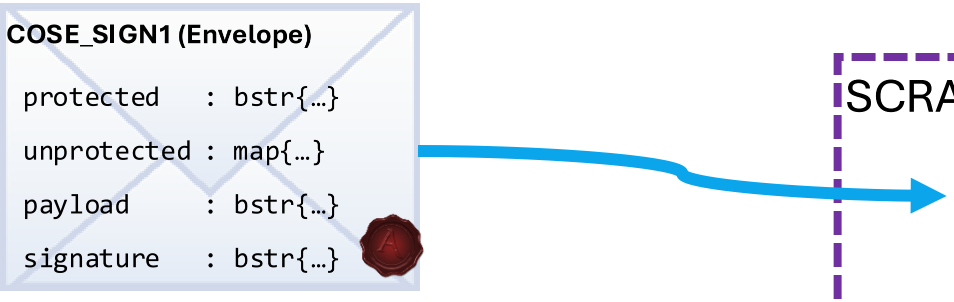
  Ancillary APIs (AAPI)

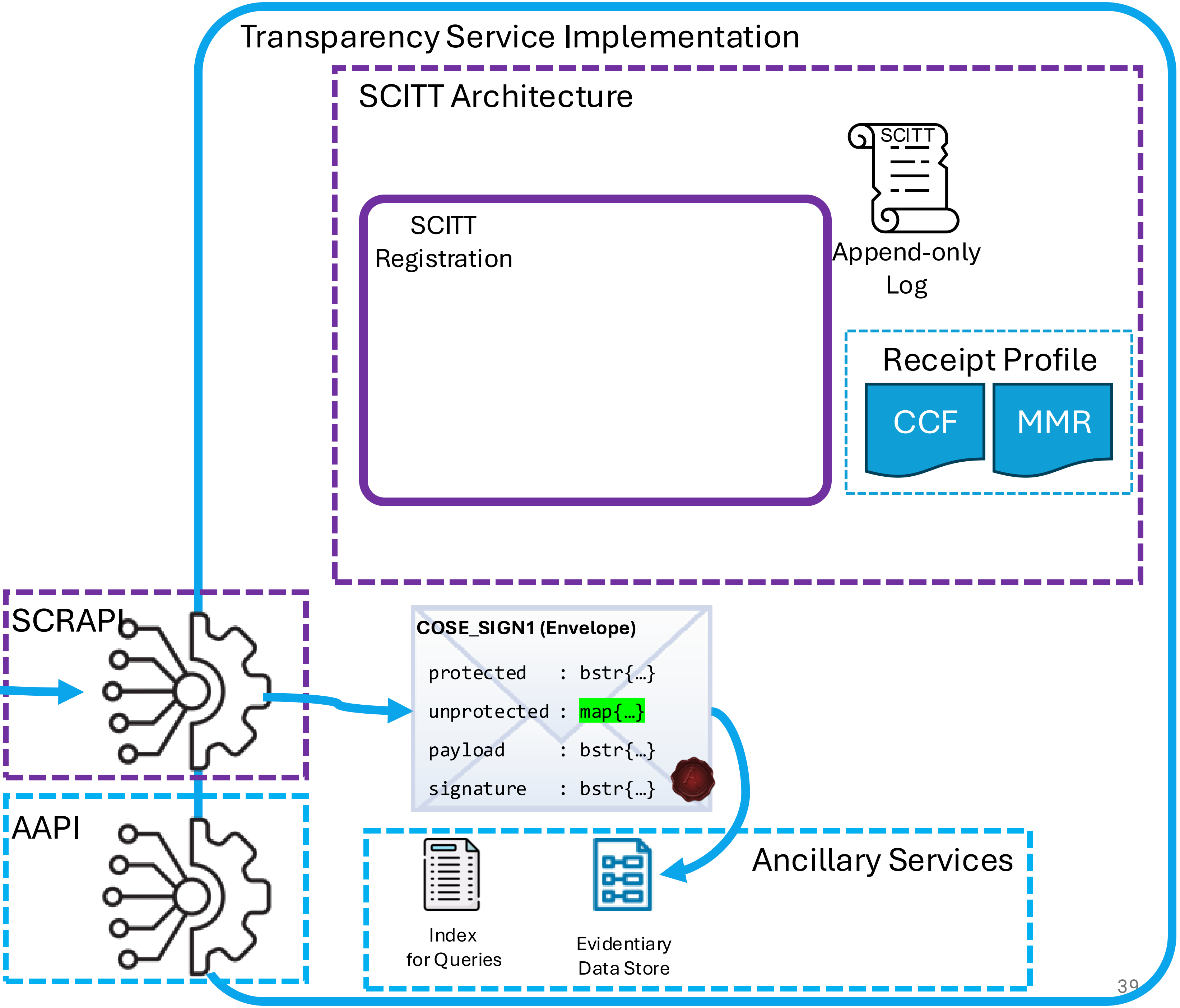


Layering

- SCRAPI
- Append-only Log
    - Receipt Profile
- Ancillary Services

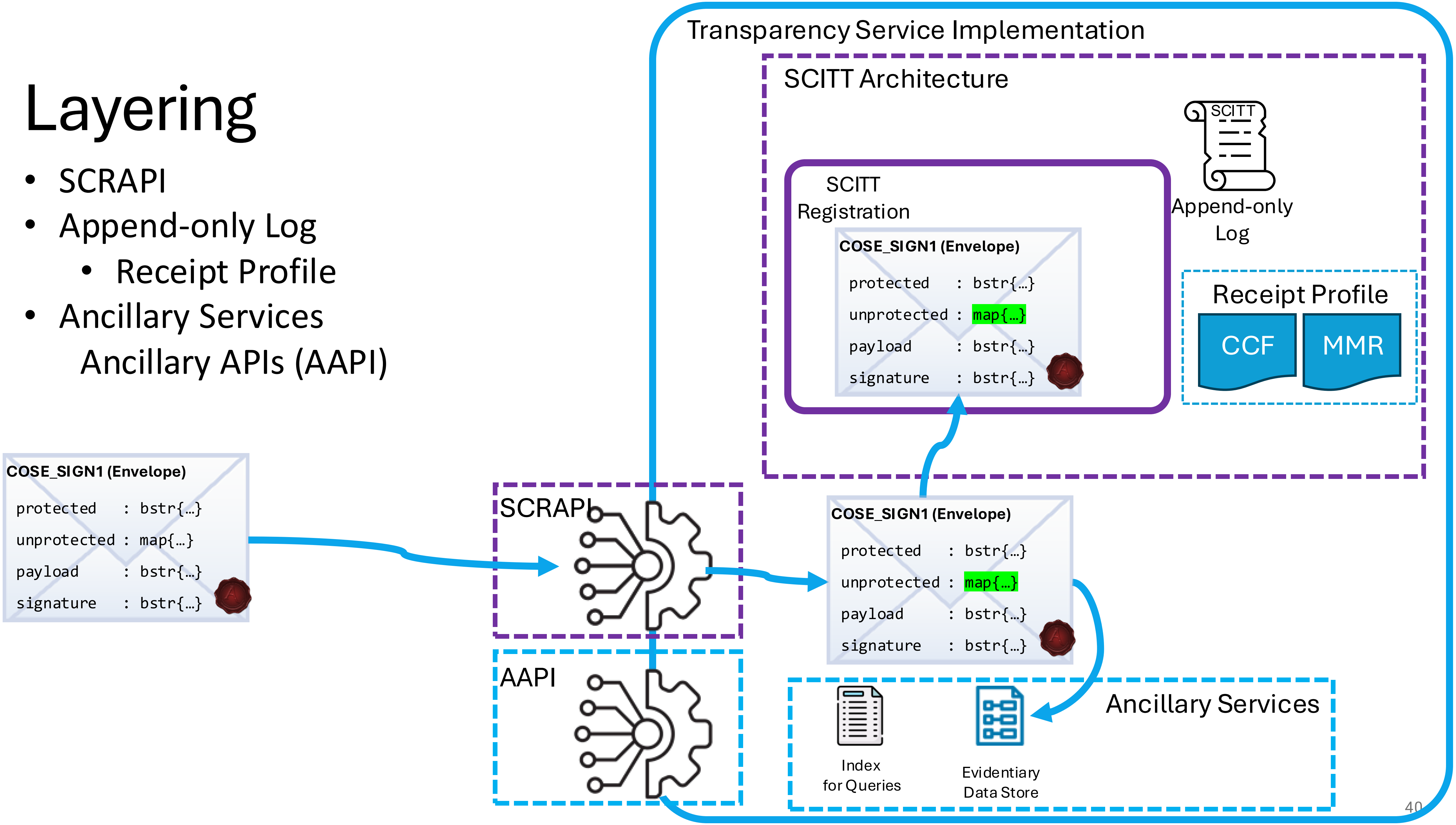
  Ancillary APIs (AAPI)





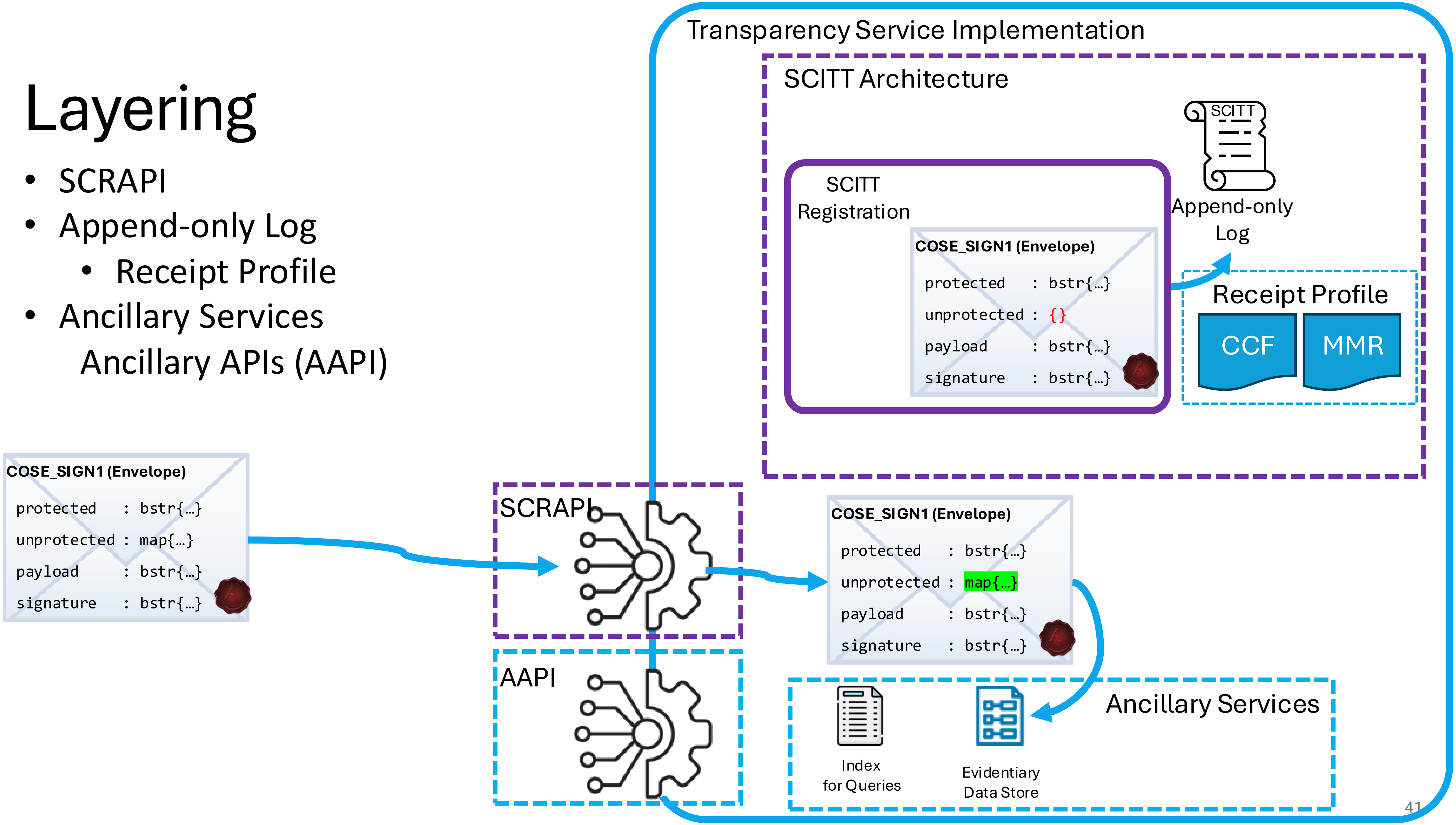
Layering

- SCRAPI
- Append-only Log
  - Receipt Profile
- Ancillary Services
  Ancillary APIs (AAPI)



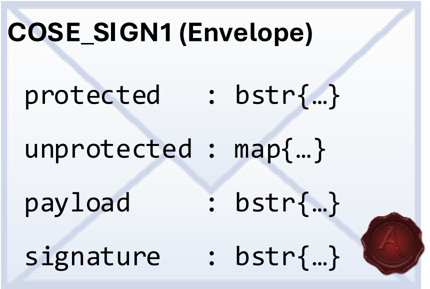
Layering

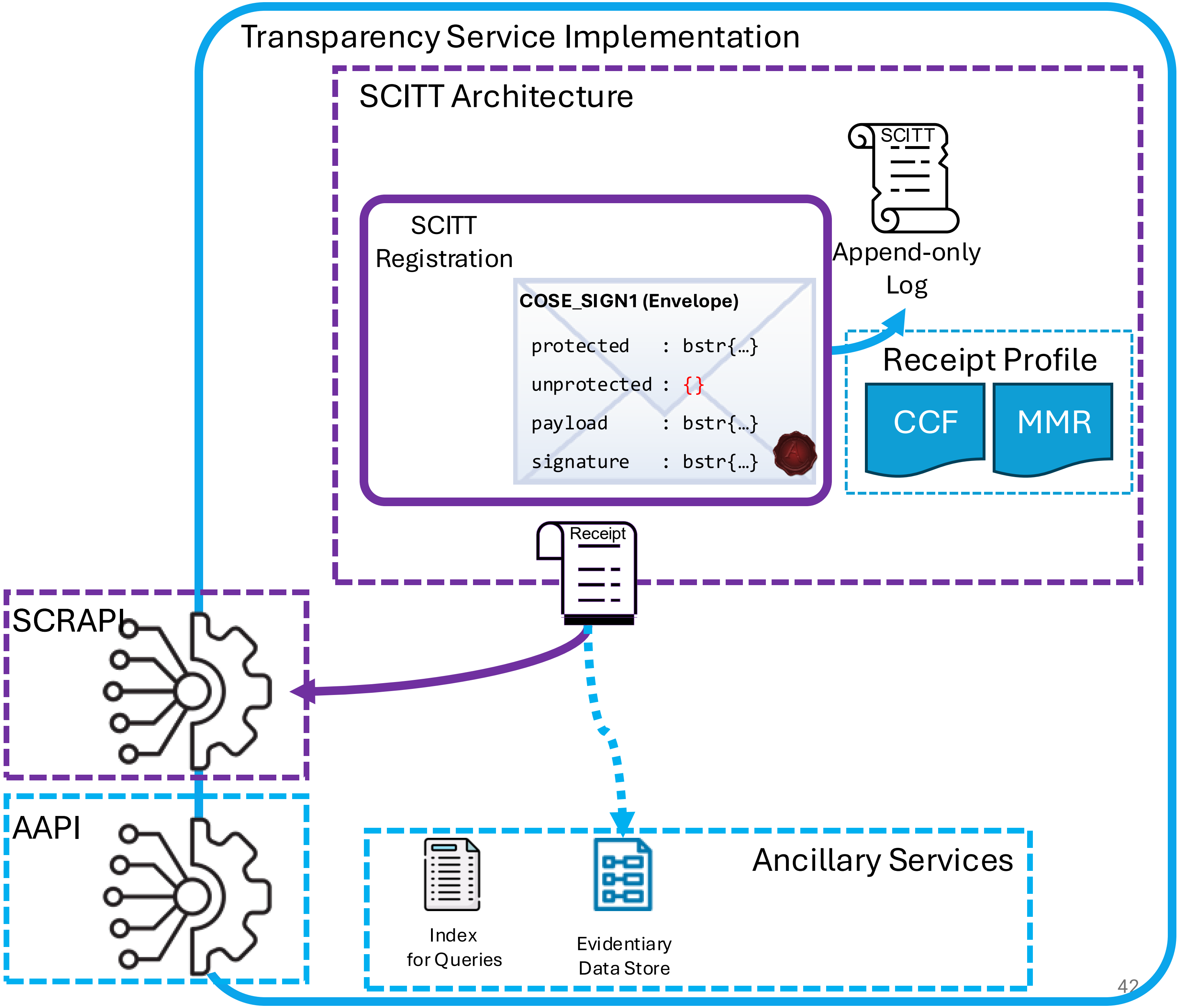
- SCRAPI
- Append-only Log
  - Receipt Profile
- Ancillary Services
  Ancillary APIs (AAPI)



Layering

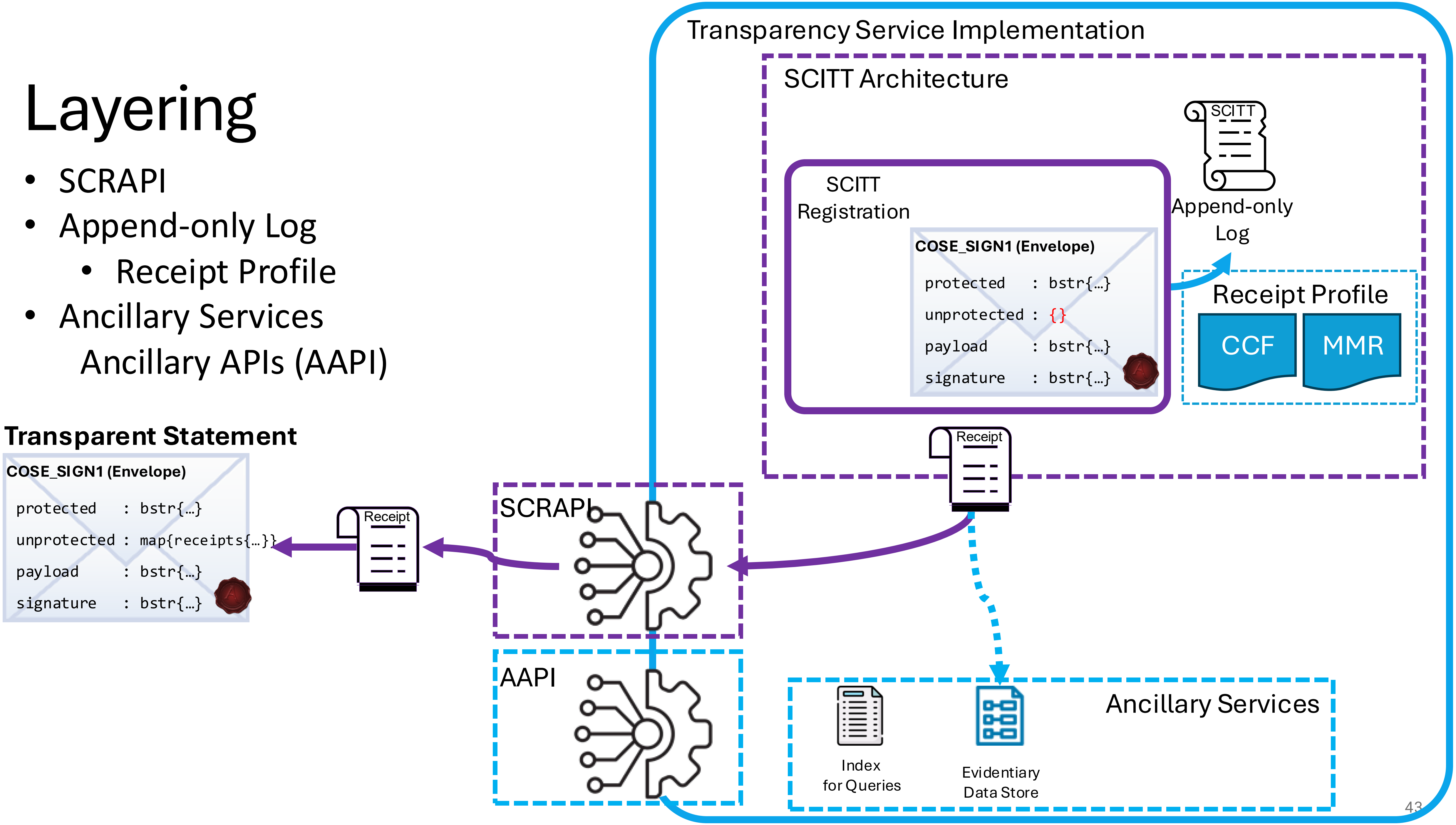
- SCRAPI
- Append-only Log
  - Receipt Profile
- Ancillary Services
  Ancillary APIs (AAPI)





Layering

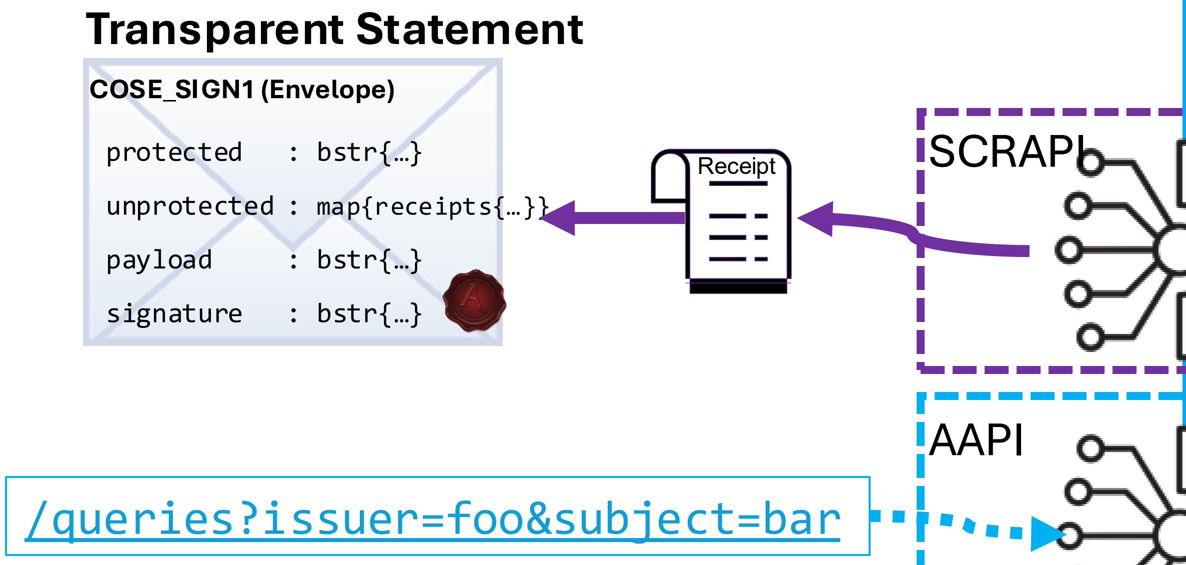
- SCRAPI
- Append-only Log
  - Receipt Profile
- Ancillary Services
  Ancillary APIs (AAPI)

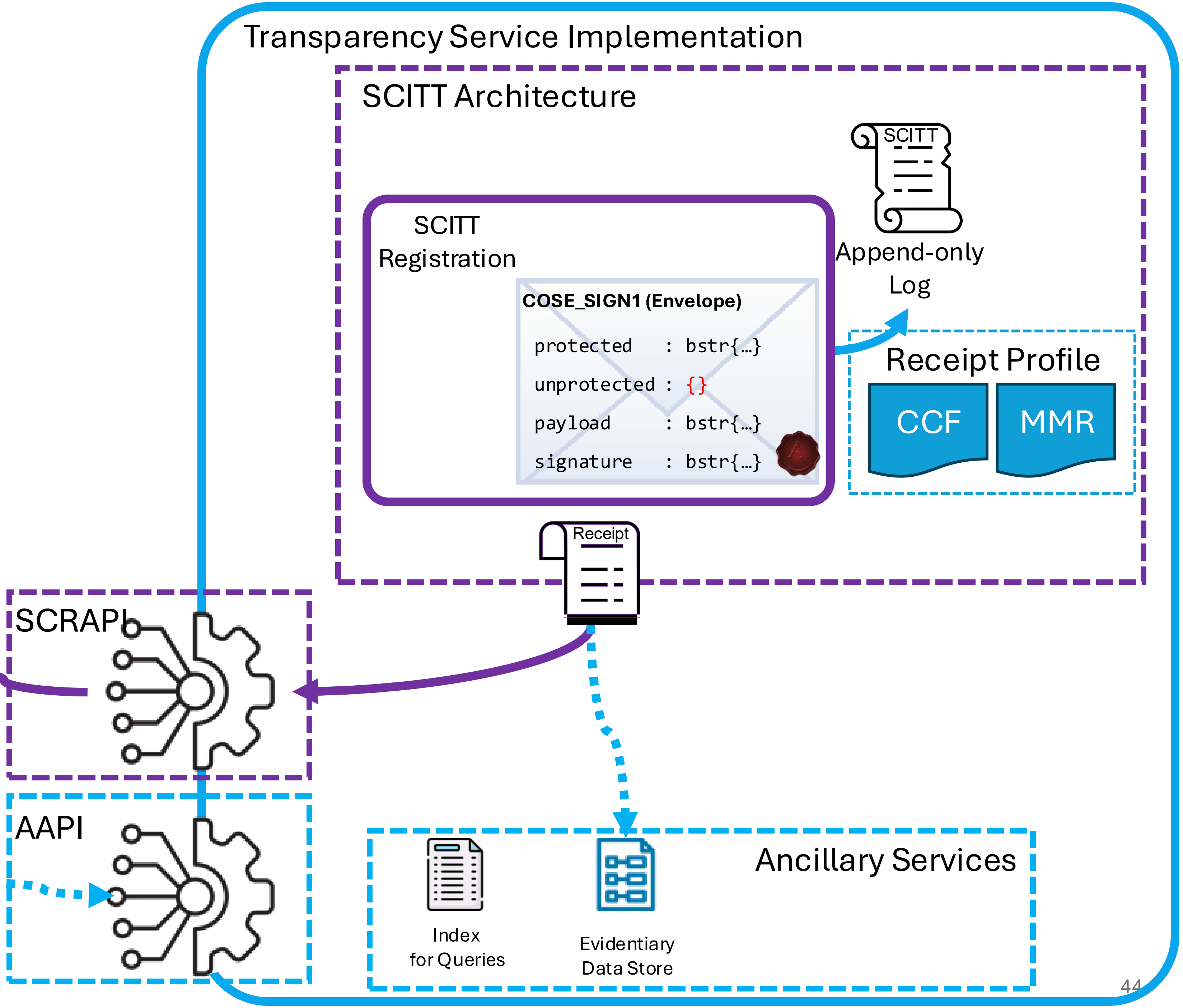


Layering

- SCRAPI
- Append-only Log
  - Receipt Profile
- Ancillary Services

  Ancillary APIs (AAPI)





SCRAPI:
SCITT Reference API

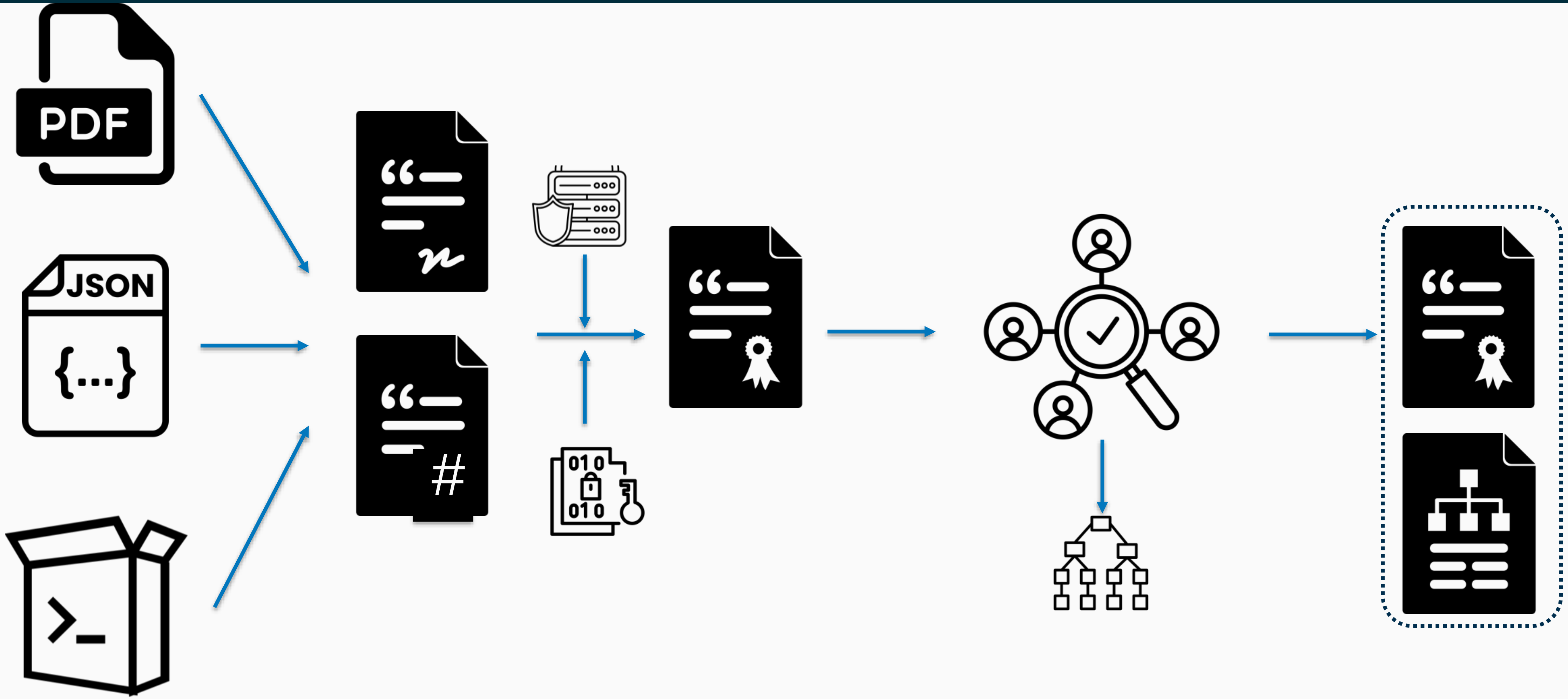
Jon Geater <jon.geater@datatrails.ai>

SCRAPI: What Does It Do?

HTTP APIs for Transparency Service-related Operations

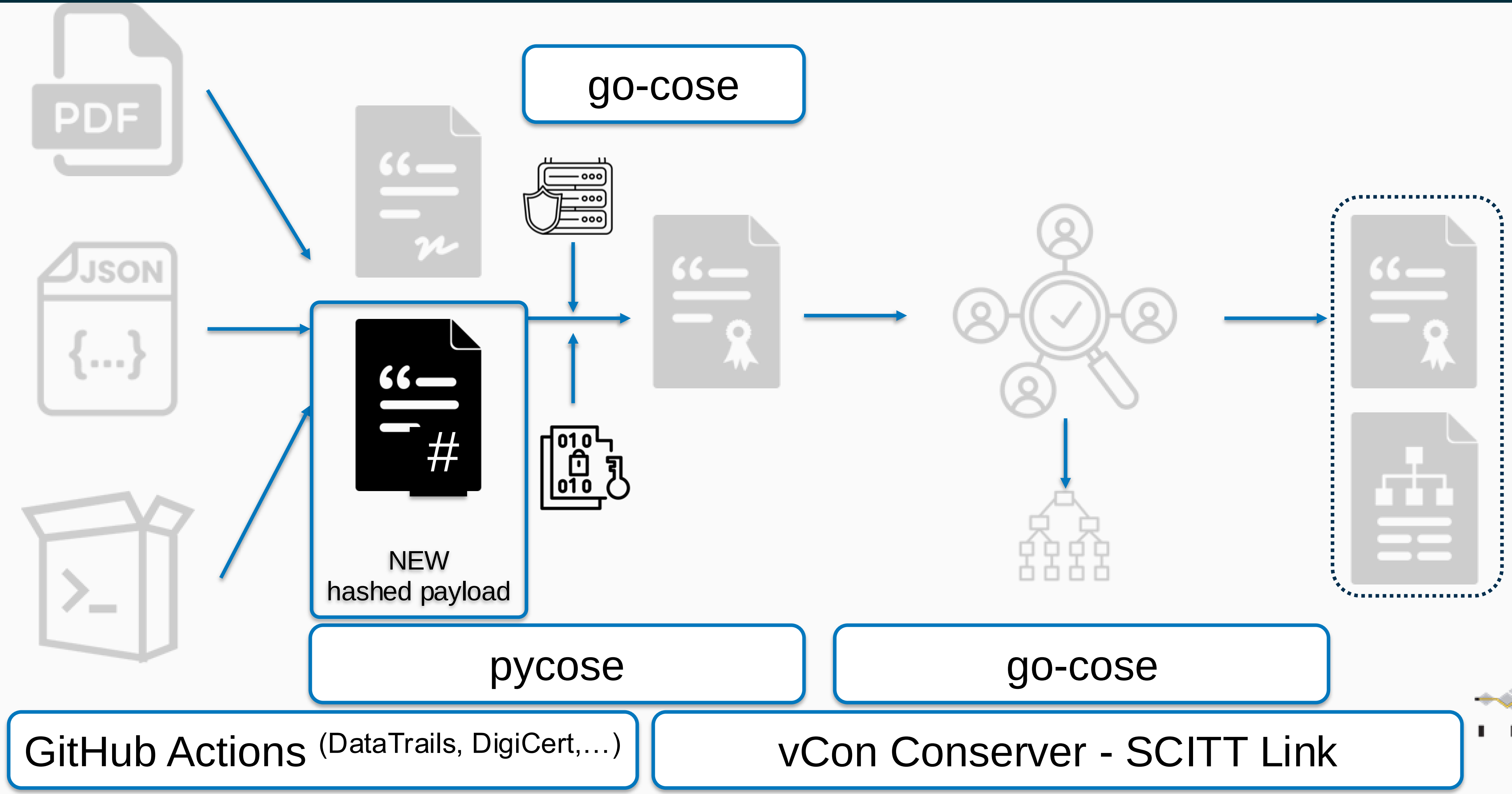
- `POST Register Signed Statement`
- `GET Receipt (Signed Proof of Inclusion for a SignedStatement)`
- `GET Signed Statements back`
- `GET Issuer Metadata`
- `GET Service Metadata (operating profile)`

Reminder from IETF 120 Hackathon … This Works!



Artifact Payload Signed Statement Transparent Statement

Reminder from IETF 120 Hackathon … This Works!



121 Issues Discussed & Resolved

- CBOR conversion related
- Error Return Types & Codes
- Long-running registrations related
- Document Flow
- Identifiers
- Resolve Statement Endpoint
- Collections of Statements
- Issue Signed Statements
- Hold The Transparency Service Accountable
- Support for Multiple Notaries
- Request Nonce Endpoint

BCP205: SCRAPI Implementations

- DataTrails /
  - scitt-community/py-scrapi
- Microsoft
  - https://github.com/microsoft/scitt-ccf-ledger/

So, We’re Done, Right?

If it worked last time, why are we still working on it?

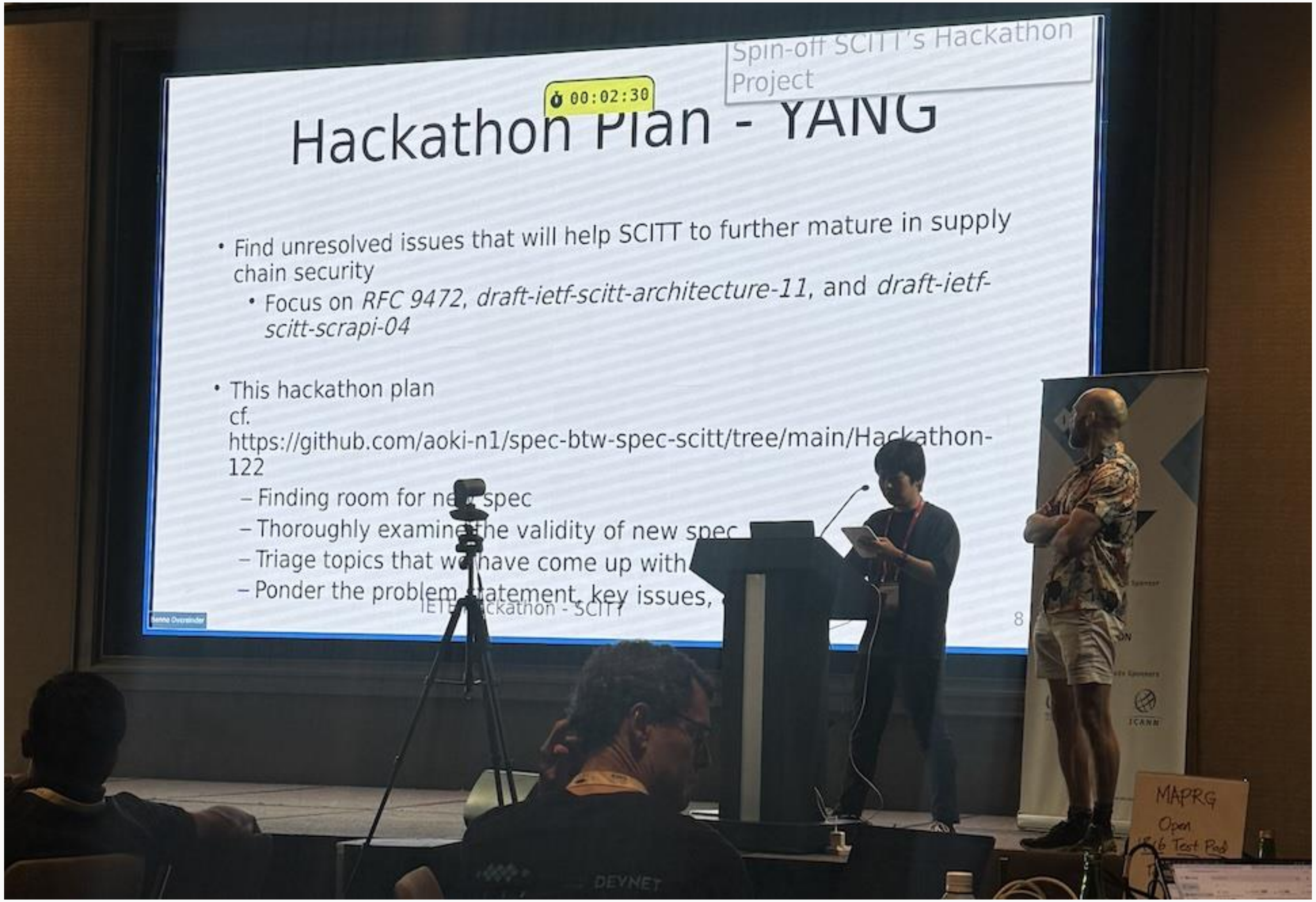
Remaining SCRAPI Issues

- Remaining are fit & finish
  - Examples to use HASH Envelope over inline content #55
  - Problems with example register signed statement #43
  - API for Collections of Statements #9

Hackathon Report

Jon Geater <jon.geater@datatrails.ai>

Team Photo



Spin-off SCITT's Hackathon Project
00:02:30
Hackathon Plan - YANG
• Find unresolved issues that will help SCITT to further mature in supply chain security
• Focus on RFC 9472, draft-ietf-scitt-architecture-11, and draft-ietf-scitt-scrapi-04
• This hackathon plan cf. https://github.com/aoki-n1/spec-btw-spec-scitt/tree/main/Hackathon-122
MAPRG

We had Two Projects

Each testing different adopted documents of the WG

Transitioning SCRAPI to HTTP-based asynchronous registration
– Jon Geater

YANG modelling of the SCITT architecture
– Nobuo Aoki

Both continue work from last time and reflect improvements in the specs since IETF 121

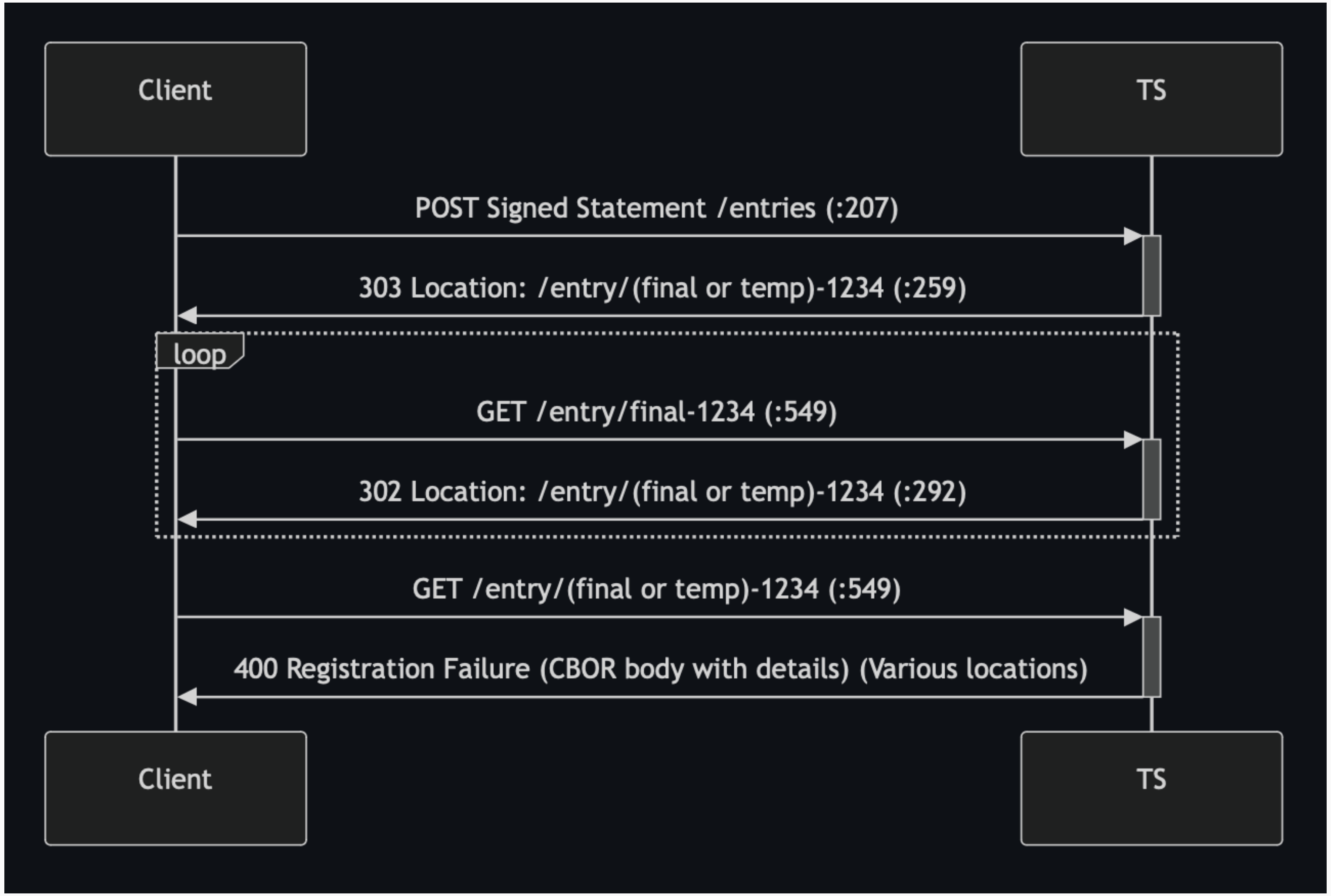
Hackathon Plan - SCRAPI

COSE/CBOR-based SCRAPI was already working well, but there have been changes to remove as many custom messages from the reference API as possible, especially where they included custom CBOR

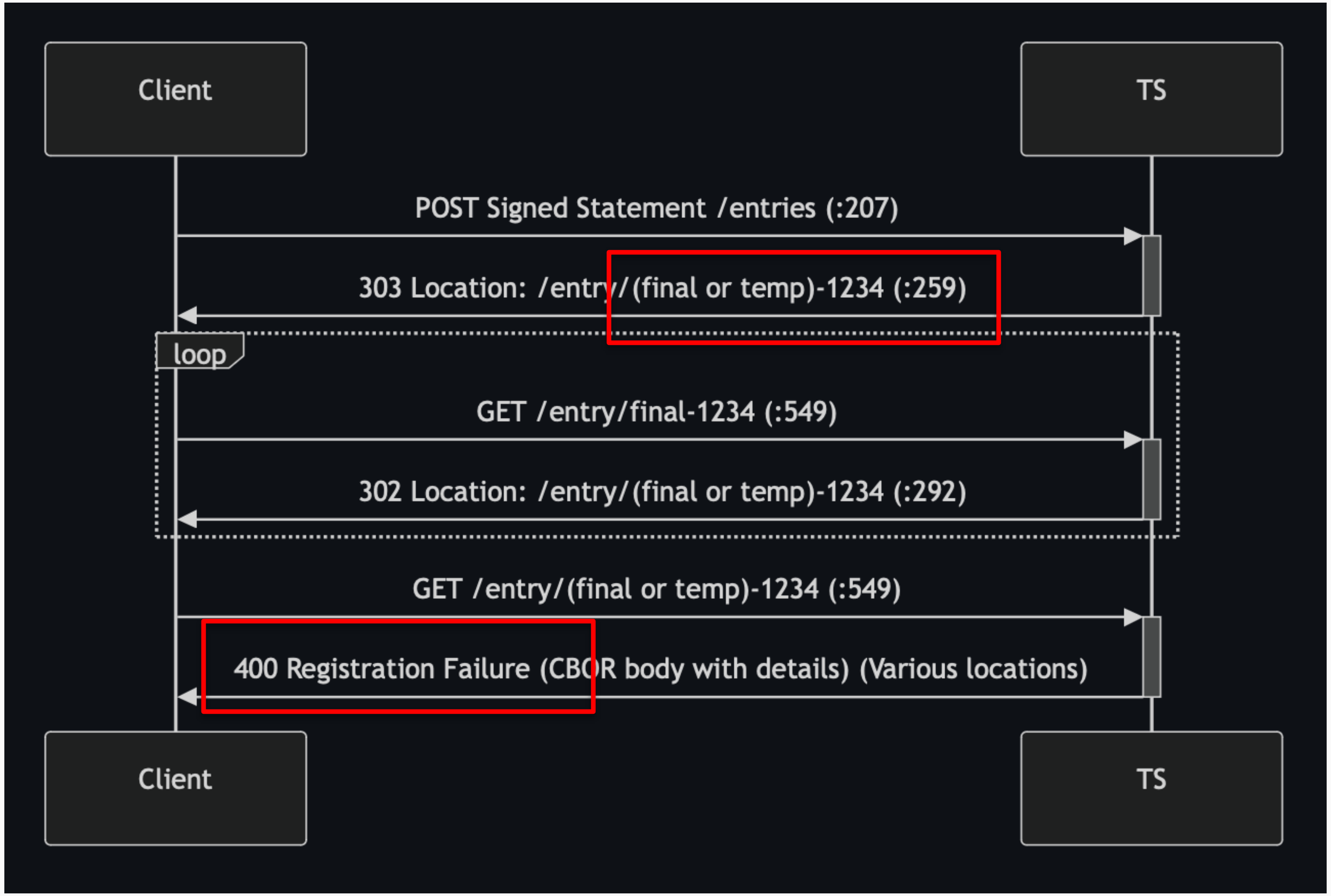
Based on this we also have a simple proof-of-concept python wrapper for the reference API which resolves everything to pycose/cbor2 and the RFC9290 objects

This hack aimed to prove that the new theoretically simpler model is also ergonomic for programmers

Concept – Why Is This Complex?



Concept – Why Is This Complex?



What Got Done - SCRAPI

What was achieved

Proof of concept transition of the former model to the new model

The same test rig passes on both!

Links:

Spec: https://github.com/ietf-wg-scitt

Code: https://github.com/scitt-community/py-scrapi

Code: https://github.com/JAG-UK/rfc9290

Releases: https://pypi.org/project/py-scrapi/

Releases: https://pypi.org/project/rfc9290/

What We Learned - SCRAPI

- The theory worked in practice!
  Only a small improvement in protocol complexity, but big improvement in code complexity 👍🏻

- Better compatibility for retro-fit implementations 👍🏻

- The REST overload and game-of-cups with locators does make the internal object handling a bit more error-prone, but it’s worth it

- Probably no new work to feed back, but will consider the opportunity and implications for very long term management of Verifiable Data Structures and whether it reads into SCITT architecture or COSE Receipts

IETF

Hackathon Plan - YANG

Spin-off SCITT's Hackathon Project

Find unresolved issues that will help SCITT to further mature in supply chain security

- Focus on RFC 9472, draft-ietf-scitt-architecture-11, and draft-ietf-scitt-scrapi-04

This hackathon plan
cf. https://github.com/aoki-n1/spec-btw-spec-scitt/tree/main/Hackathon-122

- Finding room for new spec
- Thoroughly examine the validity of new spec
- Triage topics that we have come up with
- Ponder the problem statement, key issues, and solutions

What Got Done - YANG

Spin-off SCITT's Hackathon Project

What did we achieve?

- We identified issues and processed them to be placed on the SCITT discussion
- YANG modeling of a set of statements that match the SCITT architecture
- Future SCITT's support for dynamic statement lifecycles

- What did we postpone to the hackathon in IETF Madrid?
  - Further exploration of unsolved problems in Supply Chain Security
  - Ponder a case study that will lead to feedback for SCITT

IETF

What We Learned - YANG

Spin-off SCITT's Hackathon Project

Lessons learned from this hackathon

- Identifying the tasks to be tackled at the Madrid hackathon
- Report on the problem statements, key issues, and solutions discovered for the SCITT WG
- Draft writing is a little bit intimidating for newcomers!

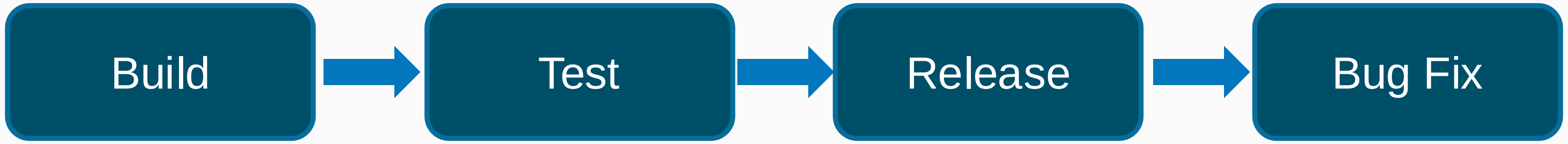
- What's next?
  - Breakdown of high-level solutions
  - Comparison of solutions that take implementation and specification descriptions into account
  - Output to some media for reporting to SCITT WG

What’s Next? - YANG

Spin-off SCITT’s Hackathon Project

How to trace complete software supply chain process?

Simple case: linear process



This one is easy: just use cwt_sub to link them together.

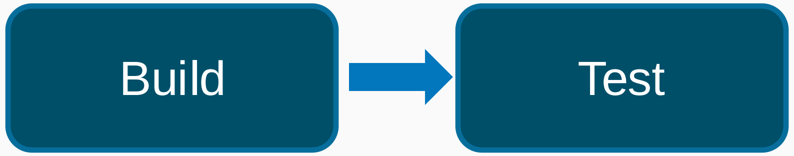
We have examples of this working for concrete artifacts and for abstract identifiers

What’s Next? - YANG

Spin-off SCITT’s Hackathon Project

How to trace complete software supply chain process?

Harder (but more likely case): messy iterative process

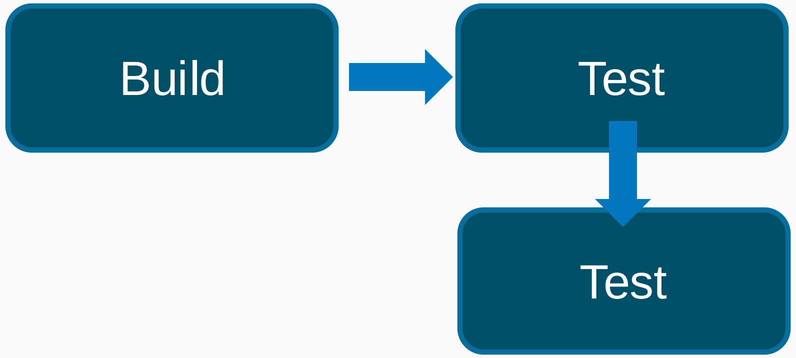


What’s Next? - YANG

Spin-off SCITT’s Hackathon Project

How to trace complete software supply chain process?

Harder (but more likely case): messy iterative process

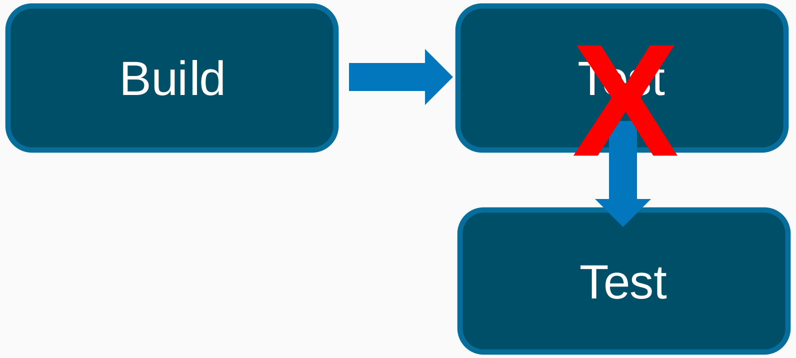


What’s Next? - YANG

Spin-off SCITT’s Hackathon Project

How to trace complete software supply chain process?

Harder (but more likely case): messy iterative process

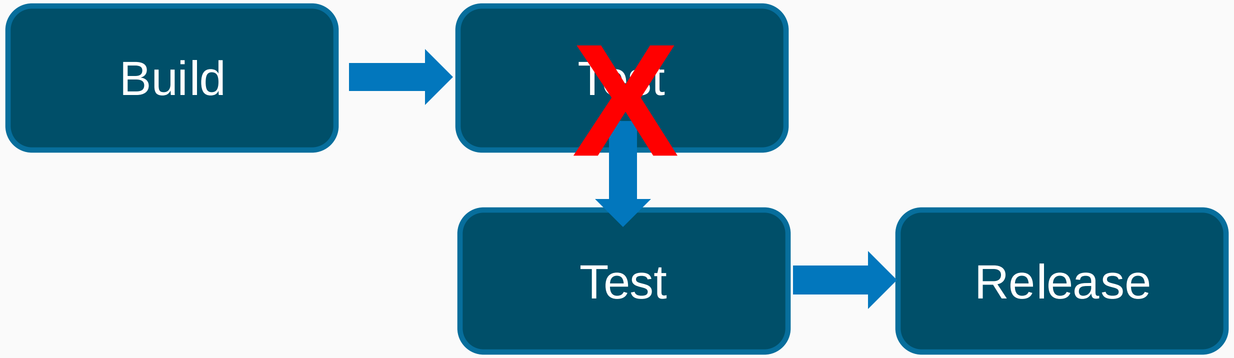


What’s Next? - YANG

Spin-off SCITT’s Hackathon Project

How to trace complete software supply chain process?

Harder (but more likely case): messy iterative process

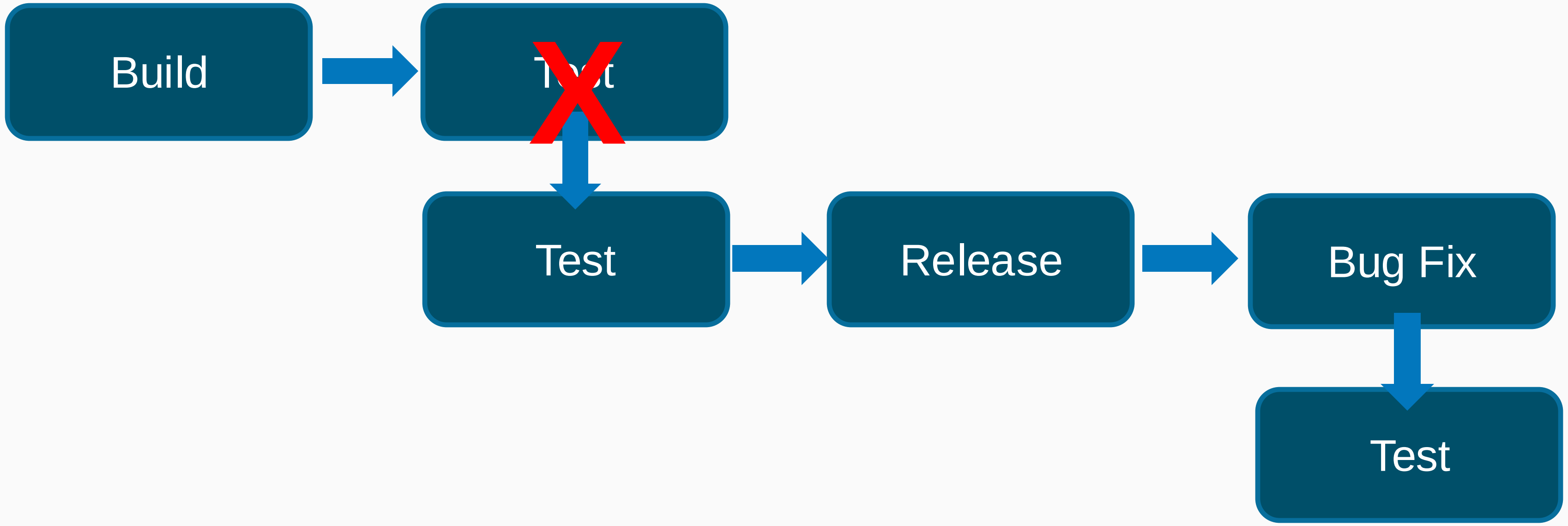


What's Next? - YANG

Spin-off SCITT's Hackathon Project

How to trace complete software supply chain process?

Harder (but more likely case): messy iterative process

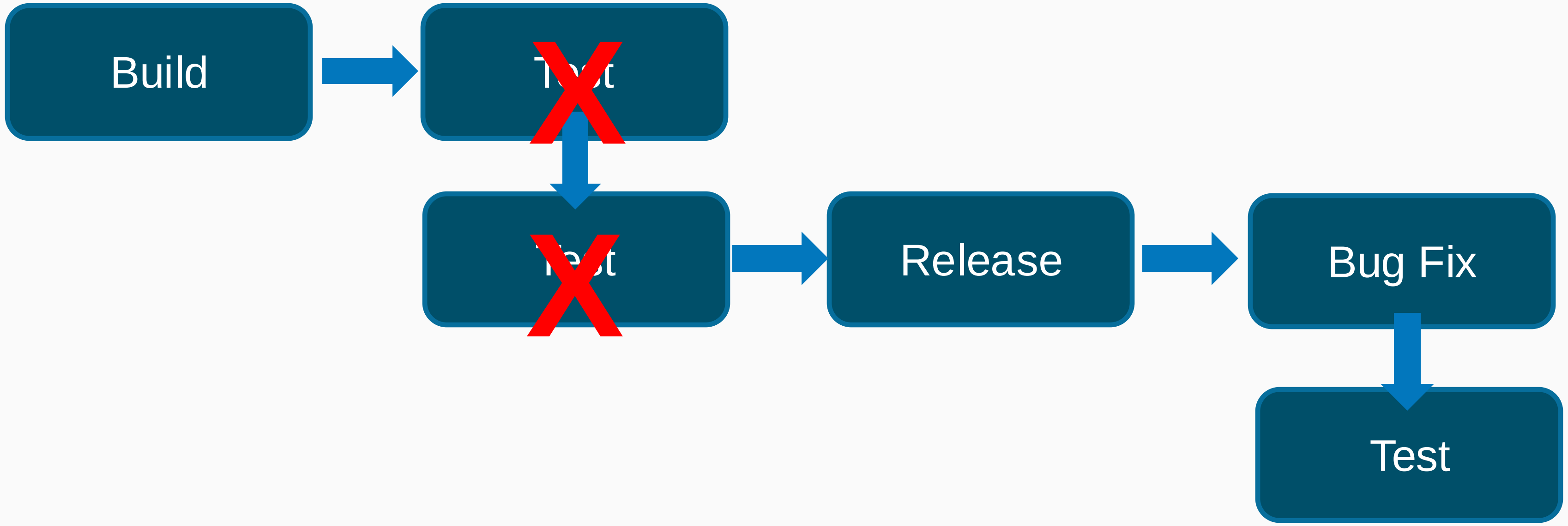


What's Next? - YANG

Spin-off SCITT's Hackathon Project

How to trace complete software supply chain process?

Harder (but more likely case): messy iterative process

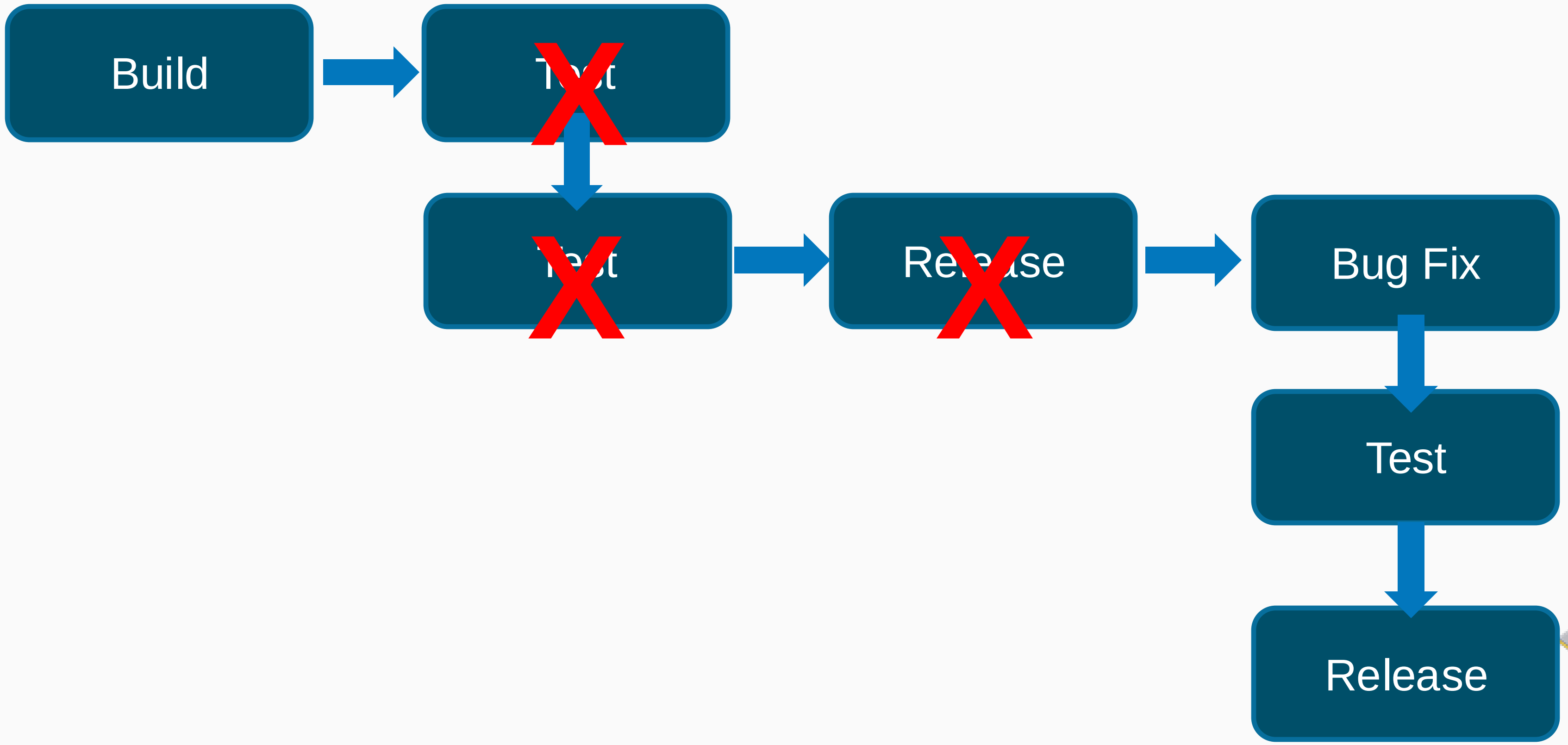


What’s Next? - YANG

Spin-off SCITT’s Hackathon Project

How to trace complete software supply chain process?

Harder (but more likely case): messy iterative process



Statements About Statements

- Is this a case of statements _about_ statements?
    - ie: this statement 'replaces' a previous one
    - as in: this test statement about release-xyz replaces the old test statement about release-xyz
- Or is it still a case of statements about an artifact that replaces another artifact?
    - I don't have good examples for this :-/

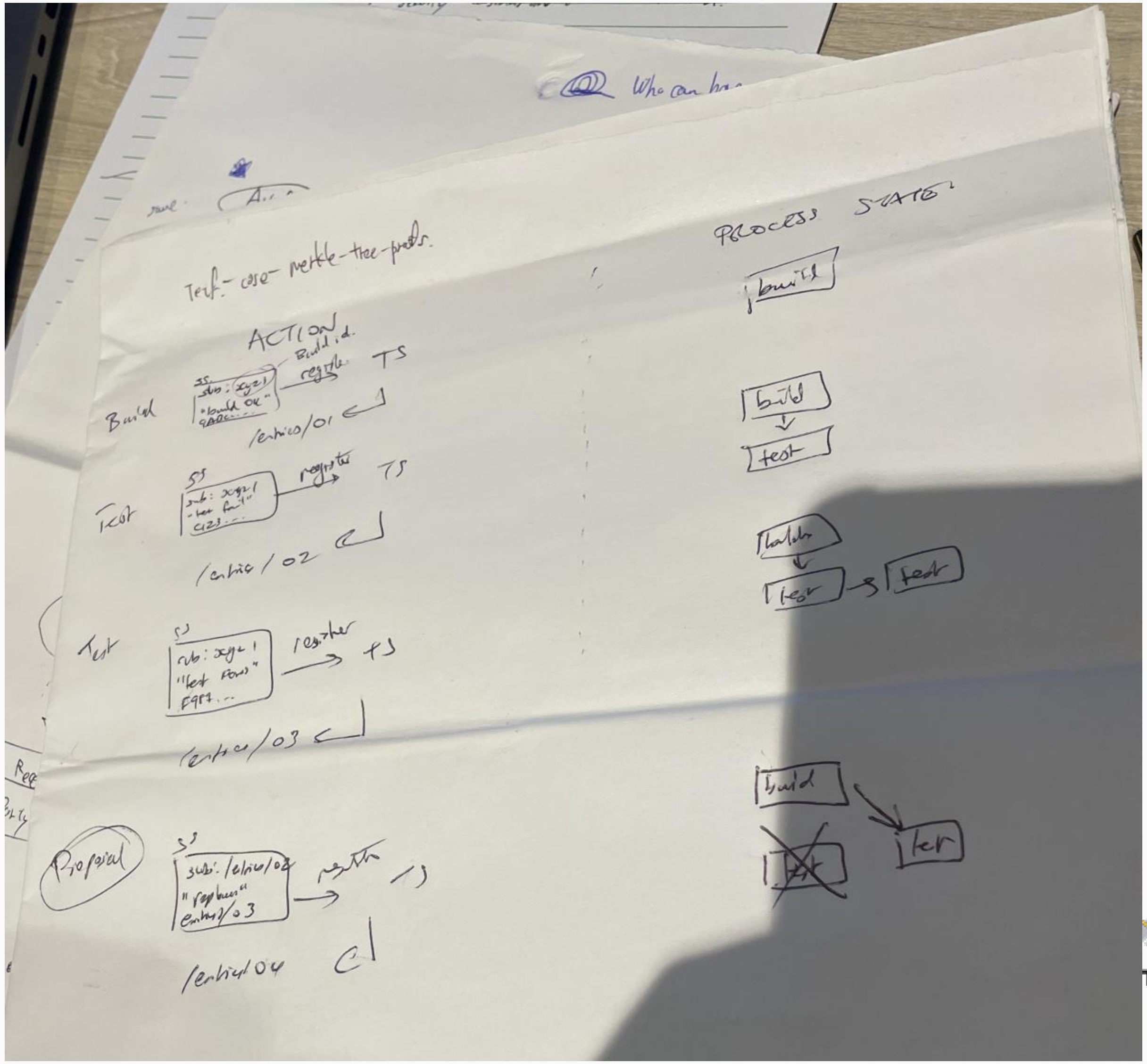
Do we need to make an architectural change to accommodate this, or is a computation one in adjacent services enough (eg fetch latest)?

This is not really any different to earlier work around 'versions' or 'sequences of statements'

Statements About Statements

This is (at least) the 3rd time statements about statements has come up as a useful and interesting concept.

This sketch seems to address the need without(?) requiring changes to the architecture, although it does seem to lean into adjacent services quite strongly.



Next Steps

Chairs

Milestone Updates

Date	Milestone	Associated documents
~~Dec 2023~~ May 2025	Submit an Architecture and Terminology document to the IESG for publication	draft-ietf-scitt-architecture
~~Dec 2023~~ √ Dec 2024	Submit a Use Cases, Security Objectives, and concise Threat Model document to the IESG for publication Merged subset into the architecture	~~draft-ietf-scitt-software-use-cases~~ draft-ietf-scitt-architecture
~~Jun 2024~~ May 2025	Submit an HTTP-based REST API for Request-Response Interactions document to the IESG for publication	draft-ietf-scitt-scrapi
Jun 2024 May 2025	Submit a Countersigning Format for Claim Registration document to the IESG for publication	draft-ietf-cose-merkle-tree-proofs
Mar 2024 <?>	Submit an Information and Interaction Model to the IESG for publication	<DISCUSSION>

From: datatracker.ietf.org/wg/scitt/about

Information Models and Interaction Models

- Registry: The WG shall define an Abstract Transparent Registry and also define the interactions and conceptual messages that will be supported by registries with an aim to generate homogeneity across multiple supply chains.
- Notarization: The WG shall develop a specification that describes the notarization information model and the interaction model a notary will use to interact with supply chain entities.
- Auditing: The WG shall develop standards to permit auditing of supply chain claims that are introduced in the transparent registry. This will, in turn, generate audit claims based on an information model (results of an audit), which can be introduced in the same registry. A corresponding interaction model will describe how audit information can be queried by supply chain consumers (e.g., end customers) before making critical business decisions.

From: datatracker.ietf.org/wg/scitt/about

SCITT Drafts

- SCITT Architecture
  https://datatracker.ietf.org/doc/draft-ietf-scitt-architecture
  - Ask: Working Group Last Call
- SCITT Reference API (SCRAPI)
  https://datatracker.ietf.org/doc/draft-ietf-scitt-scrapi/

AOB (Open Mic)

Wrap-Up

Thank you