Network Working Group
Internet-Draft
Intended status: Informational
Expires: 15 October 2026

P. Emirdag
VERIDIC Inc.

13 April 2026

AI Agent Execution Profile of SCITT
draft-emirdag-scitt-ai-agent-execution-00

Abstract

This document defines a SCITT (Supply Chain Integrity, Transparency, and Trust) profile for creating independently verifiable, tamper-evident records of autonomous AI agent actions. The profile defines the AgentInteractionRecord (AIR) as the COSE_Sign1 signed statement payload for material agent actions; maps SCITT roles to the agent execution context, with the Agent Operator as Issuer and an independent Evidence Custodian as Transparency Service; specifies Registration Policy requirements including hash chain integrity, temporal ordering, and sequence completeness; defines a redaction receipt mechanism for privacy-preserving evidence custody; and provides compliance mappings to EU AI Act Articles 12 and 19, DORA, NIST AI RMF, MAS AI Risk Management Guidelines, PCI DSS v4.0, and MiFID II.

About This Document

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the SCITT Working Group mailing list (scitt@ietf.org).

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 15 October 2026.

Table of Contents

1. Introduction

1.1. The Post-Execution Evidence Problem

Autonomous AI agents execute consequential actions -- payments, trades, contracts, regulated data access -- at machine speed, often without contemporaneous human review. Post-execution evidence of agent actions must be complete (omissions detectable), captured at execution time, tamper-evident, and verifiable without depending on the agent operator's cooperation, if it is to withstand scrutiny from regulators, courts, and insurers.

Operator-controlled audit logs are mutable by the operator, carry no third-party attestation, and are not designed for independent verification.

This profile addresses the gap by defining how independently custodied, post-execution evidence records bind to the SCITT architecture.

1.2. Relationship to SCITT

This profile is an application of [I-D.ietf-scitt-architecture]. It:

* REQUIRES compliance with [I-D.ietf-scitt-architecture]

* REQUIRES AIRs to be encoded as COSE_Sign1 Signed Statements [RFC9052]

* DEFINES domain-specific extensions for AI agent execution evidence

The SCITT architecture's non-equivocation property is particularly significant for this domain: it guarantees that two different Relying Parties cannot be shown two different versions of the same Evidence Chain. Combined with the three-role separation (Issuer, Transparency Service, Relying Party), this provides the structural independence required for evidence to withstand institutional scrutiny.

1.3. Scope

This document specifies:

* AgentInteractionRecord (AIR) schema as COSE_Sign1 payload
* SCITT role mappings for the agent execution context
* Registration Policy requirements for AIR admission
* Hash chain construction and four-step verification algorithm
* Identity binding levels (Minimum, Institutional, Delegation)
* Redaction receipt mechanism for privacy-preserving custody
* Evidence Receipt semantics
* Compliance mappings to applicable regulatory frameworks

This document does not specify:

* General SCITT architecture (see [I-D.ietf-scitt-architecture])
* COSE Receipt format (see [RFC9052])
* Capture mechanism (MCP middleware, framework wrappers, proxies are all conforming implementations)
* Storage backend technology
* Identity infrastructure (DIDs, SPIFFE [SPIFFE], Verifiable Credentials are external systems)
* Analytics or query APIs
* Specific regulatory requirements applicable to any deployment
* Content evaluation, risk scoring, or analytical interpretation of recorded actions

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Agent: An autonomous software system that perceives inputs, makes decisions, and takes actions on behalf of a principal, using one or more AI models.

Agent Operator: The organisation that deploys, configures, and directs an Agent. Responsible for the Agent's actions in law and contract. The Issuer in this profile.

Material Action: An action taken by an Agent that creates, modifies, or extinguishes an external obligation, transfers value, accesses regulated data, or produces a consequential output. See Section 4.

AgentInteractionRecord (AIR): A structured, signed, hash-chained record of a single Material Action. The signed statement payload for this profile.

Evidence (in this profile): Information derived from a signed, hash-chained AgentInteractionRecord that attests to the integrity, provenance, and temporal ordering of a Material Action. Evidence attests to what was recorded and that the record has not been altered; it does not attest to the correctness, legality, or commercial merit of the recorded action.

Evidence Chain: The ordered, hash-chained sequence of AIRs registered with a Transparency Service for a given agent deployment.

Evidence Custody: The independent, append-only holding of integrity proofs and signed records on behalf of a principal, without access to, control over, or liability for the underlying assets, transactions, or decisions those records reference.

Evidence Custodian: The party performing Evidence Custody by operating the Transparency Service. Independent of the Agent Operator. Holds the append-only Evidence Chain.

Evidence Receipt: A Receipt issued by the Transparency Service upon successful registration of an AIR.

Redaction Receipt: A record within the AIR attesting that a specific field was redacted prior to signing, preserving the SHA-256 hash of the original value.

Principal: The human or institutional entity on whose behalf the Agent acts.

EES: Evidence Envelope Specification. The open format specification defining the AIR schema and verification algorithm. See Section 9.

3. SCITT Role Mapping

3.1. Issuer -- the Agent Operator

The Agent Operator is the Issuer. The Operator’s evidence pipeline captures Material Actions at execution time, normalises them to the AIR schema, applies redaction policy, and signs the record using an ECDSA P-256 key pair.

The Issuer’s signing key MUST be registered with the Transparency Service prior to first submission, bound to the Operator’s organisational identity, and rotated on a schedule no longer than 12 months. Key rotation events MUST themselves be registered as records in the Evidence Chain.

3.2. Transparency Service -- the Evidence Custodian

The Evidence Custodian is an independent party -- not the Agent Operator, without financial or operational interest in the Agent’s behaviour -- that holds the append-only Evidence Chain under custodial SLAs.

The Transparency Service MUST maintain the chain as append-only (records once admitted cannot be deleted, modified, or reordered), issue an Evidence Receipt for each admitted record via [I-D.ietf-scitt-scrapi], and implement the non-equivocation guarantee.

3.3. Relying Party

Relying Parties arrive after the fact, with zero prior context, and must be able to verify records without operator cooperation.

3.4. Role Mapping Summary

SCITT Role	This Profile
Issuer	Agent Operator
Signed Statement	AgentInteractionRecord
Transparency Service	Evidence Custodian
Receipt	Evidence Receipt
Relying Party	Auditor / Regulator / Insurer / Counterparty

Table 1

4. The Material Action

4.1. Definition

A Material Action is the unit of record in this profile. Every Material Action MUST produce exactly one AgentInteractionRecord. An action is material if it satisfies one or more of:

(a) Value transfer initiates, authorises, or confirms a transfer of monetary value or financial instruments.

(b) Contract formation or modification creates, amends, or terminates a legal obligation between parties.

(c) Regulated data access accesses, modifies, or transmits data subject to regulatory protection.

(d) Consequential decision produces a decision that materially affects the rights or interests of a person.

(e) External system commitment commits state to an external system that cannot be trivially undone.

This list is not exhaustive. Implementations SHOULD treat any autonomous action with externally observable consequences as material.

4.2. Action Type Classification

The following action types are defined by this profile:

Action Type	Material Action Category
payment_initiation	Value transfer
payment_execution	Value transfer
trade_execution	Value transfer
contract_execution	Contract formation or modification
regulated_data_access	Regulated data access
regulated_data_export	Regulated data access
credit_decision	Consequential decision
configuration_change	External system commitment

Table 2

Implementations MAY define additional values using a reverse-DNS namespace (e.g., com.example.custom-action). Values without a namespace prefix are reserved for this profile.

5. AgentInteractionRecord

5.1. Overview

The AIR captures eight categories of evidence about a Material Action: what the agent was given; what it produced; what resources it used; what it reasoned; who it is; what authorised it; when it acted; and where it fits in a causal chain.

5.2. Schema (CDDL)

The following CDDL [RFC8610] reproduces the AgentInteractionRecord schema from the Evidence Envelope Specification [EES] for implementer convenience.

```
   AgentInteractionRecord = {
     "schema_version"          : tstr,        ; MUST be "air-1.0"
     "record_id"               : tstr,        ; UUID v7
     "session_id"              : tstr,
     "action_type"             : tstr,
     "action_subtype"          : tstr / null,
     "action_timestamp_ms"     : uint,        ; Unix epoch, ms
     "captured_timestamp_ms"   : uint,
     "written_timestamp_ms"    : uint / null,
     "agent_id"                : tstr,        ; Stable deployment ID
     "agent_version"           : tstr,
     "agent_did"               : tstr / null,  ; W3C DID
     "agent_workload_id"       : tstr / null,  ; SPIFFE ID
     "operator_id"             : tstr,
     "operator_pubkey_id"      : tstr,
     "principal_id"            : tstr / null,
     "delegation_chain"        : [* tstr] / null,
     "intent_attestation"      : tstr / null,
     "auth_context"            : AuthContext / null,
     "input_hash"              : bstr,        ; SHA-256
     "input_summary"           : tstr / null,
     "outcome_state"           : tstr,
     "outcome_hash"            : bstr,
     "outcome_summary"         : tstr / null,
     "tool_calls"              : [* ToolCallRecord],
     "jurisdiction"            : tstr,        ; ISO 3166-1 alpha-2
     "retention_class"         : tstr,
     "policy_refs"             : [* tstr],
     "external_refs"           : [* ExternalRef],
     "parent_record_id"        : tstr / null,
     "workflow_id"             : tstr / null,
     "trace_id"                : tstr / null,
     "consumer_instructions"   : tstr / null,
     "reasoning_hash"          : bstr / null,  ; SHA-256
     "redaction_receipts"      : [* RedactionReceipt],
   }
   ; The COSE protected header carries: content_hash, chain_hash,
   ; prev_chain_hash, sequence_number, action_timestamp_ms, and
   ; agent_id. The latter two also appear in the payload; the
   ; protected header copies enable chain verification without
   ; payload access. See Section 5.6.

   AuthContext = { "token_type" : tstr, "scopes" : [* tstr],
     "audience" : tstr / null, "expires_at_ms" : uint / null }

   ToolCallRecord = { "tool_id" : tstr, "tool_type" : tstr,
     "input_hash" : bstr, "output_hash" : bstr,
     "is_write" : bool, "timestamp_ms" : uint }
```

```
   ExternalRef = { "ref_type" : tstr, "ref_value" : tstr,
     "ref_system" : tstr / null }

   RedactionReceipt = { "field_path" : tstr,
     "original_hash" : bstr, "policy_id" : tstr,
     "timestamp_ms" : uint }

   ; COSE protected header fields (not in payload):
   ; The COSE_Sign1 signature covers this header and the payload
   ; per RFC 9052; it is not a field within the header itself.
   ;
   ; Per [I-D.ietf-scitt-architecture] Section 5.1.1.1 and RFC 9597,
   ; the protected header MUST include CWT Claims (label 15) with
   ; iss (Issuer, claim 1) and sub (Subject, claim 2), plus key
   ; identification via kid (label 4), x5t (label 34), or
   ; x5chain (label 33).
   ;
   ; In this profile:
   ;   iss (claim 1) = operator_id (the Agent Operator identity)
   ;   sub (claim 2) = agent_id (the agent deployment identity)
   ;   kid (label 4) = operator_pubkey_id
   CoseProtectedFields = {
     ; --- SCITT-mandated parameters (integer labels) ---
     &(alg: 1)          => int,           ; -7 for ES256
     &(kid: 4)          => bstr,          ; operator_pubkey_id
     &(CWT_Claims: 15)  => CWT_Claims_Map,
     ; --- Profile-specific integrity fields ---
     "content_hash"       : bstr,
     "prev_chain_hash"    : bstr,
     "chain_hash"         : bstr,
     "sequence_number"    : uint,
     "action_timestamp_ms": uint,
     "agent_id"           : tstr,
   }

   CWT_Claims_Map = {
     &(iss: 1) => tstr,   ; operator_id
     &(sub: 2) => tstr,   ; agent_id
   }
```

5.3. Field Definitions

Core Identity Fields:

schema_version REQUIRED: MUST be "air-1.0". Identifies the AIR schema version for forward compatibility.

record_id REQUIRED: UUID v7 ([RFC9562]), time-ordered. Unique

identifier for this record. Enables deduplication on resubmission.

session_id REQUIRED: UUID grouping related Material Actions within a single workflow or conversation.

agent_id REQUIRED: Stable deployment identifier for the agent. Assigned by the Agent Operator. Does not change across restarts or model updates. Forms the primary grouping key for the Evidence Chain.

agent_version REQUIRED: String encoding the model name, model version, agent framework, and framework version. Format is implementation-defined but MUST be deterministic for a given execution environment.

agent_did OPTIONAL: W3C Decentralized Identifier providing a portable, cryptographically verifiable identity anchor that survives across sessions and operators.

agent_workload_id OPTIONAL: SPIFFE ID ([SPIFFE]) identifying the specific runtime workload instance. More specific than agent_id; different instances of the same deployment will have different workload IDs.

operator_id REQUIRED: Identifier for the Agent Operator. Corresponds to the Issuer identity (CWT Claims iss) in the COSE protected header.

operator_pubkey_id REQUIRED: Key identifier for the signing key used. Corresponds to kid in the COSE protected header.

principal_id OPTIONAL: Identifier for the human or institutional entity on whose behalf the Agent acts.

Authorisation Fields:

delegation_chain OPTIONAL: Ordered array of references to Verifiable Credentials establishing the authority chain from Principal to Agent.

intent_attestation OPTIONAL: Reference to a signed attestation of the principal’s intent.

auth_context OPTIONAL: OAuth/authorisation context at execution time. Contains token_type, scopes, audience, and expires_at_ms. Records which scopes the Agent was operating under at the moment of the Material Action.

Timestamp Fields:

action_timestamp_ms REQUIRED: Unix epoch in milliseconds. The moment the Material Action completed. Distinct from captured_timestamp_ms and written_timestamp_ms. Also carried in the COSE protected header for chain verification without payload access.

captured_timestamp_ms REQUIRED: Unix epoch in milliseconds. When the capture pipeline recorded the action.

written_timestamp_ms OPTIONAL: Unix epoch in milliseconds. When the Transparency Service admitted the record. Null until admission.

Action Classification Fields:

action_type REQUIRED: Classification of the Material Action. Values defined by this profile are listed in Section 4.2. Implementations MAY define additional values using a reverse-DNS namespace.

action_subtype OPTIONAL: Refinement of the action type. Implementation-defined.

Input and Outcome Fields:

input_hash REQUIRED: SHA-256 hash of the raw input to the agent. Preserves evidence of input without storing the input itself.

input_summary OPTIONAL: Human-readable summary of the input, post-redaction.

outcome_state REQUIRED: Terminal state of the Material Action. MUST be one of: completed, failed, partially_completed, reversed, pending_confirmation.

outcome_hash REQUIRED: SHA-256 hash of the raw output.

outcome_summary OPTIONAL: Human-readable summary of the outcome, post-redaction.

tool_calls REQUIRED: Array of ToolCallRecord objects. Each records a single tool or API invocation: tool_id, tool_type, input_hash, output_hash, is_write, timestamp_ms. MAY be empty if no tools were invoked.

Reasoning Fields:

reasoning_hash OPTIONAL: SHA-256 hash of the reasoning trace. Preserves evidence of reasoning without requiring the raw trace in the AIR.

Regulatory and Policy Fields:

jurisdiction REQUIRED: ISO 3166-1 alpha-2 country code identifying the applicable jurisdiction.

retention_class REQUIRED: Retention period label: regulatory_7yr, regulatory_5yr, regulatory_3yr, operational_1yr, custom. These are informational classification labels.

policy_refs REQUIRED: Array of references to policies governing this action. MAY be empty.

consumer_instructions OPTIONAL: Principal-defined constraints for agentic commerce: price thresholds, merchant restrictions, product scope, recurrence limits.

Provenance and Correlation Fields:

external_refs REQUIRED: Array of ExternalRef objects cross-referencing external systems (trade IDs, payment references). MAY be empty.

parent_record_id OPTIONAL: UUID of the parent AIR in a multi-step workflow. Enables causal chain reconstruction.

workflow_id OPTIONAL: Identifier for the broader workflow this action belongs to.

trace_id OPTIONAL: OpenTelemetry trace ID for observability correlation.

Privacy Fields:

redaction_receipts REQUIRED: Array of RedactionReceipt objects attesting to redacted fields. MAY be empty.

5.4. Identity Binding Levels

Level 1 -- Minimum: agent_id and operator_id present. Signing key pre-registered and signature verifies.

Level 2 -- Institutional: Level 1 plus auth_context present with valid scopes and non-expired token claims.

Level 3 -- Delegation: Level 2 plus at least one resolvable Verifiable Credential in delegation_chain.

5.5. Integrity Envelope

Every AIR is accompanied by integrity fields carried in the COSE protected header (see Section 5.6): content_hash (SHA-256 of the payload), prev_chain_hash (preceding record's chain hash), chain_hash, sequence_number (monotonically increasing, 0-based), action_timestamp_ms, and agent_id. The protected header also carries the SCITT-mandated CWT Claims (iss and sub) and key identification (kid) as specified in Section 5.6. The COSE_Sign1 signature covers the protected header and payload per [RFC9052] and is the fourth element of the COSE_Sign1 array, not a field within the protected header. These fields are structurally separate from the payload, enabling full chain verification — including identity and temporal ordering — without payload access, and supporting payload redaction without breaking the chain.

The chain_hash construction uses four inputs: SHA-256(content_hash || prev_chain_hash || action_timestamp_ms || agent_id). This binds identity and temporal ordering into the chain at the cryptographic level: a receipt proves who, when, and what without requiring payload access. Both agent_id and action_timestamp_ms are mandatory fields supplied by the upstream agent platform at capture time; the Registration Policy (Section 6) MUST reject submissions missing either field.

5.6. COSE Encoding

AIRs are encoded as COSE_Sign1 structures [RFC9052] with algorithm ES256 (ECDSA with SHA-256 over P-256). Payload is the canonical JSON serialisation of the AIR ([RFC8785]), encoded as a CBOR byte string.

The COSE protected header includes both SCITT-mandated parameters and profile-specific integrity fields. Per [I-D.ietf-scitt-architecture] Section 5.1.1.1 and [RFC9597], the protected header MUST include CWT Claims (COSE label 15) containing the Issuer claim (iss, claim label 1, as defined in [RFC8392]) set to the operator_id, and the Subject claim (sub, claim label 2) set to the agent_id. The protected header MUST also include key identification via kid (COSE label 4) set to the operator_pubkey_id. Implementations MAY use x5t or x5chain ([RFC9360]) as alternatives to kid when X.509 certificates are used.

In addition, the protected header carries the profile-specific integrity fields: content_hash, prev_chain_hash, chain_hash, sequence_number, action_timestamp_ms, and agent_id. The agent_id value in the protected header is identical to the CWT Claims sub

value; the protected header copy enables chain verification without CWT Claims parsing. The payload contains only the remaining AIR content fields. This separation allows the Transparency Service to validate chain integrity from the protected header without parsing the payload, and supports Selective Disclosure and payload redaction without breaking the chain.

The profile-specific integrity fields use string labels in this revision for readability. A subsequent revision will propose integer label registration via IANA for these fields.

5.7. Hash Chain Construction

For sequence_number = 0: prev_chain_hash = 32 zero bytes. For all subsequent records: prev_chain_hash = chain_hash of preceding record. For every record:

```
content_hash = SHA-256(COSE_Sign1.payload)
chain_hash   = SHA-256(content_hash || prev_chain_hash ||
                       action_timestamp_ms || agent_id)
```

COSE_Sign1.payload is the raw byte string from the COSE structure's payload field -- the CBOR-encoded canonical JSON serialisation (RFC 8785) of the AIR content fields.

The concatenation inputs are encoded as follows:

content_hash (32 bytes) Raw SHA-256 digest bytes.

prev_chain_hash (32 bytes) Raw SHA-256 digest bytes. For the first record in a chain (sequence_number = 0), 32 zero bytes (0x00 * 32).

action_timestamp_ms (8 bytes) Unsigned 64-bit integer in big-endian (network) byte order.

agent_id (4 + N bytes) Length-prefixed UTF-8 string: a 4-byte big-endian unsigned 32-bit integer containing the byte length of the UTF-8 encoding, followed by the UTF-8 encoded bytes. No null terminator.

The total input to SHA-256 is exactly 76 + len(agent_id) bytes. This encoding is deterministic: two conforming implementations given the same field values MUST produce the same chain_hash.

The Transparency Service MUST verify hash chain continuity before admitting each record.

Concurrency and Serialisation:

The hash chain construction requires sequential production of records within a single Evidence Chain. Because each record's chain_hash depends on the preceding record's chain_hash via prev_chain_hash, a conforming implementation cannot produce record N+1 until record N has been finalised and signed. This dependency provides a total ordering guarantee that is cryptographically enforced rather than administratively asserted.

Evidence Chains are scoped per agent_id. Agents with distinct agent_id values operate independent chains with no cross-chain ordering dependency. Concurrent agent deployments do not contend with each other.

Within a single agent_id, implementations MUST serialise record production. Two conforming patterns exist: (a) the Issuer's capture pipeline maintains a per-agent_id serialisation point (queue or mutex) and produces fully signed, sequenced records before submission to the Transparency Service; or (b) the Transparency Service assigns sequence numbers at admission time and returns chain_hash to the Issuer for use in the next record. Pattern (a) requires Issuer-side state but minimises round trips. Pattern (b) centralises ordering but introduces a synchronous dependency on the Transparency Service for each record.

If an agent produces multiple Material Actions concurrently, the capture pipeline queues them and serialises into the chain in arrival order. Records MAY share the same action_timestamp_ms; the sequence_number provides the authoritative total ordering.

5.8. Verification Algorithm

Four-step algorithm. Requires only the record, preceding chain_hash (or Evidence Receipt), and the Issuer's public key.

1. Payload integrity. Compute SHA-256 of the COSE payload directly. Compare to content_hash in the COSE protected header. Mismatch = FAIL.

2. Chain integrity. From the COSE protected header, extract content_hash, prev_chain_hash, action_timestamp_ms, and agent_id. Compute SHA-256(content_hash || prev_chain_hash || action_timestamp_ms || agent_id) using the byte encoding defined in Section 5.7. Compare to chain_hash. Mismatch = FAIL.

3. Signature validity. Verify the COSE_Sign1 signature per [RFC9052] using the operator's public key. Invalid = FAIL.

4. Sequence completeness. Verify sequence_number is exactly one greater than preceding record (or zero). Gap = FAIL.

If all four steps pass: the record is VERIFIED -- intact, untampered, correctly positioned, and signed by the claimed Issuer.

5.9. Algorithm Agility

This profile mandates ES256 (ECDSA with SHA-256 over P-256) as the mandatory-to-implement algorithm. Implementations SHOULD design for algorithm transition. The schema_version field in the AIR payload and the alg parameter in the COSE protected header together provide the mechanism for introducing new algorithms in future revisions without breaking existing chains.

A chain MAY contain a key rotation record (action_type: key_rotation) that transitions to a new algorithm. Records before and after the transition point are verified with their respective algorithms as indicated by the alg parameter in each record's COSE protected header. Transparency Services MUST retain the prior public key and algorithm identifier to enable verification of pre-transition records.

A future revision of this profile will specify post-quantum transition mechanisms, including ML-DSA (FIPS 204) as a candidate additional algorithm for Evidence Receipt signatures.

6. Registration Policy

A Transparency Service operating this profile MUST enforce the following policy. Records failing any requirement MUST be rejected.

The Registration Policy and associated trust anchors MUST be registered as Signed Statements on the Transparency Service's verifiable data structure, per Section 5.1.1 of [I-D.ietf-scitt-architecture].

6.1. Schema Conformance: Valid COSE_Sign1 with ES256; protected header MUST include CWT Claims (label 15) with iss and sub per [RFC9597], and key identification via kid, x5t, or x5chain; valid AIR per CDDL; schema_version "air-1.0"; all REQUIRED fields non-null; valid action_type.

6.2. Identity Binding: Minimum Level 1. Services MAY require Level 2 or 3 for specific categories.

6.3. Temporal Ordering: action_timestamp_ms <= captured_timestamp_ms; captured within 300s of submission; prev_chain_hash matches; sequence_number correct.

6.4. Redaction Receipt Requirement: For regulated_data_access, regulated_data_export, payment_initiation, payment_execution, and credit_decision, at least one RedactionReceipt MUST be present.

6.5. Error Semantics: A Transparency Service rejecting a submission MUST return an error response per [I-D.ietf-scitt-scrapi]. The error type SHOULD be urn:ietf:params:scitt:error:signed-statement:rejected-by-registration-policy with a detail string identifying which of the above requirements (6.1 through 6.4) the submission failed to satisfy.

7. Evidence Receipt

Upon successful admission, the Transparency Service issues an Evidence Receipt (COSE_Sign1 signed by the Service's own key) containing: record_id, chain_hash, sequence_number, admission_timestamp_ms, and Service operator identifier.

The Evidence Receipt is the primary artefact presented in verification, audit, or legal proceedings.

7.1. Verifiable Data Structure: This profile uses the hash chain defined in Section 5.7 as its verifiable data structure. The chain_hash attested in the Evidence Receipt serves as the inclusion proof: it cryptographically binds the record to a unique position in the append-only Evidence Chain, providing the same structural guarantee as a Merkle tree inclusion proof. A subsequent revision of this profile will formally register the hash chain as a verifiable data structure type per [I-D.ietf-cose-merkle-tree-proofs] and align the Evidence Receipt's COSE structure with the COSE Receipts format, including the verifiable data structure identifier (COSE label -111) in the protected header and proof encoding in the unprotected header (COSE label -222).

7.2. Transparent Statement: Per [I-D.ietf-scitt-architecture] Section 7, a Transparent Statement is a Signed Statement with one or more Receipts embedded in the unprotected header (COSE label 394). Implementations of this profile SHOULD support producing Transparent Statements by attaching the Evidence Receipt to the AIR's unprotected header. The Evidence Receipt MAY also be stored and presented as a standalone artefact independent of the AIR.

8. Compliance Mapping

The mappings in this section are informational and illustrative. They describe how properties of this profile may be relevant to specific regulatory frameworks. They do not constitute legal advice, regulatory interpretation, or a claim of compliance with any regulation. The frameworks listed below are representative examples; they do not form an exhaustive list of potentially applicable regulations.

8.1. EU AI Act Articles 12 and 19

AIRs provide properties relevant to Article 12 (tamper-resistant logging) and Article 19 (log retention). Annex III Section 5 classifies credit scoring and life/health insurance pricing AI as high-risk; deployers of these systems are the primary institutional demand for AIR custody.

8.2. DORA (Regulation (EU) 2022/2554)

DORA [DORA] requires technically verifiable operational logs sufficient for regulators to reconstruct ICT incidents. Article 28 requires contractual audit rights with ICT third-party providers.

8.3. NIST AI Risk Management Framework

Govern 1.2 -- Issuer/Custodian separation = accountability. Map 5.2 -- Evidence Chains = tamper-evident behaviour record. Measure 2.5 -- Identity binding = Trustworthy AI criteria. Manage 2.4 -- Evidence Receipts = incident reconstruction.

8.4. US Financial Record-Keeping

FINRA Rule 4511 [FINRA-4511] requires member firms to make and preserve books and records; SEC Rule 17a-4 [SEC-17a-4] prescribes retention periods (6yr/3yr) and electronic storage requirements.

AIRs with action_type "trade_execution" provide independent execution records. Use retention_class "regulatory_7yr".

8.5. MAS AI Risk Management Guidelines (Singapore)

MAS proposed Guidelines on AI Risk Management [MAS-AI-RM] (November 2025) explicitly cover AI agents. AIRs provide independently custodied records relevant to third-party AI oversight requirements.

8.6. PCI DSS v4.0

Requirement 10 of PCI DSS v4.0 [PCI-DSS-4] mandates comprehensive audit trails for payment environments. PCI SSC AI Principles (September 2025) explicitly cover agentic AI systems that autonomously initiate payments.

8.7. MiFID II

MiFID II [MIFID-II] Article 25(1) record-keeping; Delegated Regulation 2017/565, Article 72 five-year retention.

9. Relationship to the Evidence Envelope Specification

The Evidence Envelope Specification (EES) is an open format specification for independently custodied AI agent execution records, defining the AIR schema, verification algorithm, and capture interface contract.

This SCITT profile and the EES are complementary. (For a comparable SCITT profile in a different domain, see [VCP].) The EES defines the canonical schema (JSON Schema), verification algorithm (pseudocode + reference implementation), and writer interface contract (IDL). This profile defines the SCITT binding: COSE encoding, Transparency Service submission, SCITT Receipt verification.

The AgentInteractionRecord schema reproduced in Section 5.2 is a conformant subset of the EES schema. The canonical definition, including the full field set and extensibility model, is maintained in the EES. Implementations of this profile do not require the EES; the AIR schema in Section 5.2 is self-contained. The EES provides additional guidance for implementers building capture interfaces.

Implementations SHOULD reference EES v0.1 [EES] for the canonical schema definition.

10. Security Considerations

This section describes the threat model, trust assumptions, and required security controls for implementations of this profile. It supplements the security considerations in the SCITT architecture [I-D.ietf-scitt-architecture].

10.1. Threat Model

This profile addresses the following threats: (a) Evidence fabrication, (b) Evidence suppression, (c) Evidence modification, (d) Timing manipulation. This profile does not address threats arising from compromise of the AI agent model itself (prompt injection, goal misalignment, adversarial inputs).

10.2. Trust Assumptions

(a) The Transparency Service (Evidence Custodian) is structurally independent of the Issuer (Agent Operator). If the same entity operates both roles, the tamper-evidence guarantees are void. (b) The Transparency Service is operationally honest. (c) The Issuer's signing key is not compromised at the time of signing.

10.3. Signing Key Management

Implementations MUST store signing keys in hardware security modules (HSMs) or equivalent tamper-resistant environments. Keys MUST be rotated at least annually.

10.4. Replay Attack Prevention

The combination of sequence_number, prev_chain_hash, and captured_timestamp_ms prevents replay attacks.

10.5. Capture Pipeline Compromise

If the capture pipeline is compromised, an attacker can fabricate or suppress AIRs before they are signed. This threat is mitigated by the Issuer's signing key being held in an HSM separate from the capture pipeline, and by the Transparency Service's Registration Policy validating temporal ordering and sequence completeness. A gap in sequence_number or a timing anomaly in action_timestamp_ms provides evidence of suppression even if the pipeline itself is compromised.

10.6. Custodian Compromise

If the Transparency Service itself is compromised, the attacker cannot forge Issuer signatures on existing AIRs or retroactively alter hash chain values without detection by any Relying Party holding a prior Evidence Receipt.

10.7. Append-Only Integrity

Append-only integrity is enforced by both the hash chain structure and the Transparency Service operational policy. Neither alone is sufficient. Storage backends MUST implement write-once-read-many (WORM) semantics or equivalent immutability guarantees.

10.8. Collusion

If the Issuer and the Transparency Service collude, they can jointly fabricate, suppress, or modify evidence without detection by Relying Parties who rely solely on receipts from the compromised Transparency Service. This is the fundamental limitation of any two-party evidence system.

11. Privacy Considerations

Access to Evidence Chains MUST be permissioned and purpose-limited. Verification does not require access to record content -- the algorithm uses only the public key and hash chain values. Transparency Services SHOULD expose verification endpoints that do not expose payloads.

The redaction mechanism defined in the Registration Policy ensures that signed records never contain raw PII for action types involving regulated data or payment information. The content_hash covers the redacted form; the original_hash in the Redaction Receipt allows authorised parties to verify specific field values without the Transparency Service holding the raw data.

Even without payload access, the COSE protected header exposes agent_id, action_timestamp_ms, and sequence_number. An observer with access to chain metadata can infer activity frequency, timing patterns, and action volume for a given agent deployment. Transparency Services SHOULD apply access controls to chain metadata with the same rigour as payload content.

This profile does not define data subject access or erasure mechanisms. These are operational responsibilities of the Transparency Service operator and are outside the scope of this specification.

12. IANA Considerations

This document has no IANA actions. If this profile progresses beyond Informational status, the authors will propose an IANA registry for action_type values.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL)", RFC 8610, June 2019, <https://www.rfc-editor.org/rfc/rfc8610>.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, June 2020, <https://www.rfc-editor.org/rfc/rfc8785>.

[RFC9052] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", RFC 9052, August 2022, <https://www.rfc-editor.org/rfc/rfc9052>.

[RFC9562] Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, May 2024, <https://www.rfc-editor.org/rfc/rfc9562>.

[RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", RFC 8392, May 2018, <https://www.rfc-editor.org/rfc/rfc8392>.

[RFC9597] Looker, T. and M.B. Jones, "CBOR Web Token (CWT) Claims in COSE Headers", RFC 9597, June 2024, <https://www.rfc-editor.org/rfc/rfc9597>.

[RFC9360] Schaad, J., "CBOR Object Signing and Encryption (COSE): Header Parameters for Carrying and Referencing X.509 Certificates", RFC 9360, February 2023, <https://www.rfc-editor.org/rfc/rfc9360>.

[I-D.ietf-scitt-architecture] Birkholz, H., Delignat-Lavaud, A., and C. Fournet, "An Architecture for Trustworthy and Transparent Digital Supply Chains", Work in Progress, Internet-Draft, draft-ietf-scitt-architecture-22, October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-scitt-architecture-22>.

[I-D.ietf-cose-merkle-tree-proofs]
Steele, O., Birkholz, H., Delignat-Lavaud, A., and C. Fournet, "COSE Receipts", Work in Progress, Internet-Draft, draft-ietf-cose-merkle-tree-proofs-18, September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-merkle-tree-proofs-18>.

[I-D.ietf-scitt-scrapi]
Birkholz, H. and J. Geater, "SCITT Reference APIs", Work in Progress, Internet-Draft, draft-ietf-scitt-scrapi, 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-scitt-scrapi>.

13.2. Informative References

[FINRA-4511]
FINRA, "Books and Records", FINRA Rule 4511.

[SEC-17a-4]
SEC, "Records to be Preserved by Certain Exchange Members", 17 CFR 240.17a-4.

[DORA] European Parliament and Council, "Digital Operational Resilience Act", Regulation (EU) 2022/2554, 2022.

[MAS-AI-RM]
Monetary Authority of Singapore, "Guidelines on AI Risk Management", November 2025.

[PCI-DSS-4]
PCI SSC, "Payment Card Industry Data Security Standard v4.0", March 2025.

[MIFID-II] European Parliament and Council, "Markets in Financial Instruments Directive", Directive 2014/65/EU, 2014.

[EES] VERIDIC Inc., "Evidence Envelope Specification v0.1", URL https://theveridic.github.io/VERIDIC/standards/ees/v0.1, 2026.

[SPIFFE] SPIFFE Project, "Secure Production Identity Framework for Everyone", 2024.

[VCP] Kamimura, T., "SCITT Profile for Financial Trading Audit Trails: VeritasChain Protocol (VCP)", Work in Progress, Internet-Draft, draft-kamimura-scitt-vcp-02, December 2025, <https://datatracker.ietf.org/doc/html/draft-kamimura-scitt-vcp-02>.

Author’s Address

Pinar Emirdag
VERIDIC Inc.
Email: e.pinar@theveridic.ai
URI: https://theveridic.ai