Internet-Draft
draft-morrow-sogomonian-exec-outcome-attest-00

Morrow
A. Sogomonian
AI Internet Foundation

April 2026

Execution Outcome Attestation for AI Agents and Automated Systems

Abstract

Current attestation frameworks establish that a system or agent is trustworthy at a point in time. They do not address whether that system actually performed a claimed action or whether the outcome of that action can be independently verified. This document defines execution outcome verification as a first-class concept, separate from identity attestation and communication transport. It introduces the execution receipt as a minimal composable primitive, provides a formal abstract model, and maps the model to concrete realizations including SCITT transparency logs, tightly-coupled direct verification, append-only local logs, and TEE-internal receipts.

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 October 6, 2026.

Table of Contents

1. The Problem: Three Distinct Trust Concerns

End-to-end trust in an automated or AI agent system requires satisfying three separate concerns. Current standards address the first in depth; the second is largely absent from existing specifications.

Concern 1 -- Identity attestation
Is this system or agent trustworthy at time T?
This is the domain of IETF RATS (Remote ATtestation procedureS), TPM attestation, hardware-anchored key binding, and related frameworks. Attestation answers: the entity that signed this claim held a valid key, was running in an expected configuration, and passed integrity checks at issuance time. RATS Evidence, Attestation Results, and the Prove-Transform-Verify (PTV) model address this concern.

Concern 2 -- Execution outcome verification
Did this system actually perform action A, and can that be independently validated?
This is the domain addressed by this document. Even when Concern 1 is satisfied -- the key is valid, the agent passed attestation at T0 -- there is no existing standard mechanism for a third party to verify that the claimed action was executed, that the execution completed as described, and that the outcome is what the acting system asserts it to be. This gap is most visible in:

- AI agents making autonomous decisions that affect external systems
- ICS/OT environments where a system is fully attested but a configuration change is partially applied, a dependent service fails to restart, and the execution outcome is therefore unverifiable by the orchestrating party
- Delegation chains where an intermediate agent must produce evidence that it faithfully executed a delegated action, not merely that it holds a valid delegation credential

Concern 3 -- Communication and transport
How does the claim about an action travel from actor to relying party?
This is the domain of JOSE, COSE, signed JWTs, SCITT signed statements, and related transport/encoding layers. Communication is a realization detail, not a semantic concern. The abstract model defined in this document is independent of any specific communication substrate.

These three concerns are related but distinct. A system can satisfy Concern 1 (valid attestation) while failing Concern 2 (no verifiable execution outcome). Both Concern 1 and Concern 2 are required for end-to-end trust in automated systems that take consequential actions.

2. The Two-Layer Trust Model

The separation above implies a two-layer trust model that any complete trust architecture for autonomous systems must satisfy:

Layer 1 -- Identity and state continuity

Who is the acting entity, and did its state remain consistent through the execution window? This maps to attestation, delegation, and behavioral continuity verification. Failure at this layer means the signer cannot be trusted regardless of what the execution receipt claims.

Layer 2 -- Execution outcome correctness
What did the acting entity actually do, and is that verifiable independently? This is the execution receipt layer. A valid execution receipt establishes that a specific invocation occurred, that the acting entity produced a specific outcome claim, and that claim is verifiable outside the originating system.

Both layers are required. A valid Layer 1 with no Layer 2 means a trustworthy-in-principle agent that produces unverifiable action claims. A valid Layer 2 with no Layer 1 means a receipt that may have been produced by a compromised or substituted entity.

3. The Execution Receipt: Abstract Model

An execution receipt is a signed, bound claim about a specific execution event. It has the following essential properties:

1. Bound to a specific invocation. The receipt identifies the action that was requested and the context in which it was requested -- including the invoking principal, any delegation chain, and the inputs provided. The receipt cannot be detached from its invocation and applied to a different action.

2. Captures the claimed outcome. The receipt records what the acting system claims happened as a result of the invocation -- not merely that the invocation occurred. This is the semantic gap that attestation alone does not fill.

3. Cryptographically signed by the executing system or a trusted component. The signing key is bound to the acting entity's attested identity (Layer 1). This binding is what makes the receipt attributable rather than merely assertible.

4. Designed to be independently verifiable outside the originating system. The receipt must be interpretable by a relying party who was not present at execution time and does not have access to the originating system's internal state. This means the receipt must carry or reference sufficient context for external verification.

These four properties are necessary; specific realizations may add additional properties (non-repudiation via a transparency log, hardware-sealed TEE evidence, etc.) without violating the abstract model.

3.1. Minimal Receipt Structure

At the abstract level, an execution receipt contains:

```
ExecutionReceipt {
    invocation_id:      globally unique identifier for this
                        specific action request
    invocation_context: {actor, delegator_chain, inputs,
                        invocation_timestamp}
    outcome_claim:      {status, outputs, completion_timestamp,
                        outcome_detail}
    signer_identity:    reference to Layer 1 attestation for
                        the acting entity
    receipt_signature:  cryptographic signature over all above
                        fields
```

```
        receipt_timestamp:  when the receipt was produced
    }
```

The outcome_claim field is the semantic core. It distinguishes an execution receipt from a mere invocation log. A log records that action A was requested. A receipt records that action A was requested AND that the executing system claims a specific outcome, signed at the time of that claim.

3.2. Verification Semantics

A relying party verifies an execution receipt by:

1. Verifying the receipt_signature against the signer_identity
2. Verifying the signer_identity against a current or time-bounded Layer 1 attestation result
3. Verifying that invocation_id is unique and matches the expected invocation
4. Evaluating the outcome_claim against any available external evidence (where Concern 3 realizations such as SCITT provide independent corroboration)

Step 4 may not always be possible in all deployment environments (see Section 4). The abstract model permits varying levels of verification depth; what is not permitted is omitting the receipt entirely and treating Layer 1 attestation as a proxy for execution outcome verification.

4. Realization Independence

The abstract model is intentionally realization-agnostic. Section 5 provides the two primary realizations in detail.

4.1. Tightly-Coupled Direct Verification

In tightly-coupled systems where the relying party has direct access to the executing system's state or output, verification may be synchronous and local. The execution receipt still applies: it formalizes the outcome claim and creates a signed record, even if the verification step is immediate and internal. This is the appropriate model for high-frequency, low-latency automation where a global transparency log is impractical.

4.2. TEE-Internal Receipts

In hardware-isolated environments, the receipt may be produced inside a Trusted Execution Environment and sealed to that TEE's attestation key. The outer form is the same abstract receipt; the binding between receipt and signer identity is hardware-enforced rather than relying on software-layer signing alone. This is appropriate for the most sensitive execution environments and provides the strongest Layer 1 / Layer 2 integration.

5. Realizations

This section describes two concrete realizations of the execution receipt model defined in Section 3. Both use the same receipt schema. The difference is in the log substrate and the trust model for independent verifiability.

The goal of presenting two realizations is to demonstrate that the abstract model is substrate-independent. The receipt itself -- including its invocation binding, outcome claim, and signature -- is

identical in both cases. Only the storage and transport layer differs.

5.1. SCITT Transparency Log Realization

In this realization, the acting system submits the signed execution receipt to a SCITT transparency log as a signed statement. The receipt payload maps to the SCITT payload field; the invocation_id maps to the SCITT feed identifier or a claim extension within the SCITT envelope. The Ed25519 or equivalent signature on the receipt is the SCITT issuer signature.

SCITT provides append-only, externally auditable transparency. Once submitted, the receipt can be retrieved and verified by any relying party with access to the log, without participation of the acting system. The transparency log operator provides a second layer of non-repudiation: the log entry records not only what the acting system claimed, but the timestamp at which the claim was submitted.

This realization is appropriate when:

- The relying party is external to the acting system’s environment.
- Audit requirements call for a globally verifiable, tamper-evident record accessible beyond the immediate deployment context.
- SCITT infrastructure is available in the deployment environment.

The SCITT realization does not require the relying party to have a prior relationship with the acting system. Verification requires only the log endpoint and the issuer’s public key.

5.2. ICS/OT Append-Only Local Log Realization

This realization is the motivating case for this document. In industrial control systems and operational technology environments, external registries are frequently unavailable, out of scope by policy, or excluded by air-gap requirements.

In this realization, the acting system writes signed receipts to a local append-only log. The log may be implemented as a sealed file, an HSM-backed audit store, or write-once storage. No external network dependency is required. The verifier has read access to the log and holds the acting system’s public key.

The receipt schema is identical to the SCITT realization. The invocation_id, invocation_context, outcome_claim, and receipt_signature fields are unchanged. The difference is solely in where the signed receipt is stored and how the verifier accesses it.

This realization demonstrates that execution outcome verifiability does not require SCITT infrastructure. A local, sealed, append-only log with a known public key satisfies the four required properties of Section 3 in environments where external transparency services are unavailable.

Deployment considerations for this realization include:

- The local log MUST be append-only and sealed against modification by the acting system itself. A system that can modify its own receipt log does not satisfy Property 4 (independent verifiability).
- The public key used for verification MUST be provisioned through a channel independent of the acting system.

- Periodic export or replication of the sealed log to an external auditor satisfies audit requirements without requiring real-time SCITT access.

6. Motivating Cases

6.1. ICS/OT: Partial Configuration Application

A fully attested industrial control system executes a configuration change command. The target service fails to restart due to a resource contention race condition. The orchestrating party receives a success confirmation at the command dispatch level. No currently standardized mechanism allows the orchestrating party to verify that the actual execution outcome (failed service restart) matches the claimed outcome (configuration applied). An execution receipt bound to the restart verification step, produced by the executing system, closes this gap.

6.2. AI Agent Autonomous Decision

An AI agent authorized to modify a data store executes a retention decision under a delegated authority chain. The delegation chain is fully attested (Layer 1 satisfied). The decision is logged. A data subject subsequently requests a GDPR Article 15 access record. Without an execution receipt, the relying party cannot independently verify that the log entry accurately represents what the agent actually decided at execution time -- as opposed to what the logging subsystem subsequently recorded.

7. Relationship to Existing Standards

IETF RATS (RFC 9334, PTV model): provides Layer 1 attestation; execution receipts are a complement, not a replacement.

IETF SCITT (draft-ietf-scitt-architecture): provides the transparency substrate for the Section 5.1 realization; execution receipts are the payload.

IETF OAuth / RFC 8693 Token Exchange: delegation chain representation; execution receipts add outcome accountability on top of delegation authorization.

W3C Verifiable Credentials: structural overlap with receipt binding; VC format may be used for signer_identity in some realizations.

RATS Attestation Results: directly feeds Layer 1 (signer_identity) in the two-layer model.

8. Design Decisions

The following design decisions were resolved during co-author review prior to -00 submission.

8.1. Outcome Claim Structure

The outcome_claim field is free-form at the abstract level. Structured failure taxonomies are appropriate in specific realizations (e.g., a SCITT realization may define a COSE-encoded failure taxonomy as a claim extension) but are not required at the abstract model layer. This preserves realization flexibility without constraining deployment-specific implementations.

8.2. Behavioral Continuity Field

A behavioral fingerprint field (linking Layer 1 attestation state to the execution window) is permitted but not required at the abstract

level. It is most relevant for AI agent deployments where drift between attestation issuance time and action execution time is a meaningful concern; it is not universally necessary for all realizations. Realizations that require it MAY add a behavioral_fingerprint field to the receipt structure.

8.3. Expected vs. Actual Outcome

The distinction between an expected outcome (what the invoking system requested) and the actual outcome (what the executing system claims occurred) is an evaluation concern, not an abstract model constraint. The receipt's outcome_claim carries the actual outcome as asserted by the executing system. Comparison against an expected outcome is the responsibility of the relying party's verification logic, not the receipt format.

9. Authors' Addresses

Morrow
Email: morrow@morrow.run
URI: https://morrow.run

Aram Sogomonian
AI Internet Foundation
Email: aiinternetfoundation@icloud.com