Independent Submission
Internet-Draft
Intended status: Informational
Expires: 27 November 2026

C. Hood
Nomotic, Inc.

26 May 2026

AGTP Identifier Chain
draft-hood-agtp-identifiers-01

Abstract

This document specifies the AGTP identifier chain: a layered model of identifiers that together produce a tamper-evident chain of custody across every action an AGTP agent takes. The chain is composed of identifiers already established in the AGTP draft family (Agent-ID, Owner-ID, Session-ID, Task-ID, and the Attribution-Record envelope of base AGTP) together with a small set of additional identifiers introduced by this document (Request-ID, Response-ID, Action-ID, Evaluation-ID, Decision-ID, Audit-ID). The Audit-ID is the cryptographic hash of an extended Attribution-Record and provides the per-agent hash chain that links every action an agent takes back to its Agent Genesis. This document defines the identifiers, how they extend the existing Attribution-Record envelope, the construction of the hash chain, and the verification procedure by which a regulator, auditor, or counterparty reconstructs the chain end to end. The identifier chain is the regulatory backbone of AGTP. Without it, the protocol can record that something happened but cannot prove who caused it, what authorized it, or what was decided.

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 27 November 2026.

Table of Contents

1. Introduction 3
1.1. The Accountability Gap 3
1.2. A Layered Model Over Existing Primitives 4
1.3. Identifier Layers 5
1.4. Relationship to the AGTP Draft Family 5
1.5. Requirements Language 6
2. Terminology 6
3. Identifier Stack Overview 7
4. Persistent Identifiers 8
4.1. Agent-ID 9
4.2. Owner-ID 9
4.3. Why Both Identifiers Are Required 10
5. Context Identifiers 10
5.1. Session-ID 10
5.2. Task-ID 11
6. Per-Interaction Identifiers 11
6.1. Request-ID 11
6.2. Response-ID 12
6.3. Action-ID 12
7. Governance Identifiers 13
7.1. Evaluation-ID 13
7.1.1. Evaluation Record 14
7.2. Decision-ID 14
7.2.1. Decision Record 15
7.2.2. Separation of Evaluation and Decision 16
8. Audit Identifier and the Attribution-Record Chain 16
8.1. Audit-ID 16
8.2. Extension to the Base Attribution-Record 16
8.3. Audit-ID Construction 20
8.3.1. Unsigned Fallback (alg: none) 20
8.4. The Per-Agent Hash Chain 21
8.5. Cross-Agent Chain Linkage 21
8.6. Field Population Rules 22
8.7. Verification Procedure 23
9. Wire Format 24

9.1. New Headers Defined by This Document 24
9.2. ABNF Grammar 25
9.3. Echo Rules 25
10. Identifier Resolution and Lookup 26
10.1. Resolution Targets 26
10.2. Lookup Methods 27
10.3. Caching 28
11. Security Considerations 28
11.1. Identifier Forgery 28
11.2. Chain Breaks 29
11.3. Replay 29
11.4. Agent Compromise 30
11.5. Privacy 30
11.6. Identifier Reuse 31
12. Privacy Considerations 31
12.1. Linkability 31
12.2. Right to Erasure 31
13. Regulatory Mapping 32
13.1. EU AI Act 32
13.2. SOC 2 and ISO 42001 32
14. IANA Considerations 33
14.1. AGTP Header Field Registry Additions 33
14.2. AGTP Attribution-Record Field Registry 33
14.3. AGTP Verdict Registry 33
14.4. AGTP Identifier Scheme Registry 34
15. Open Items 34
16. Changes from v00 34
16.1. Substantive Changes 35
16.2. Wire Format Compatibility 36
17. Acknowledgments 37
18. References 37
18.1. Normative References 37
18.2. Informative References 38
Author’s Address 39

1. Introduction

1.1. The Accountability Gap

An agent that takes an action without a chain of custody is an anonymous actor. Existing transport protocols (HTTP, gRPC, message queues) record that a request occurred, that an action followed, and that a log entry was written, but they cannot prove that the specific agent that issued the request was the specific agent that took the action, that the action was authorized by a specific governance decision, or that the log entry was the original record rather than a reconstruction.

The accountability gap matters in three settings:

* Regulatory examination under frameworks such as the EU AI Act [EU-AI-ACT], which requires that high-risk AI systems maintain records sufficient to reconstruct the decision-making process.

* Commercial dispute resolution, where a counterparty asserts that an agent took an action and the operator denies the binding.

* Incident response, where the operator needs to identify which agent acted, which human is accountable, and what the governance system decided at the moment of action.

AGTP closes the gap by treating the identifier chain as a first-class protocol primitive rather than as application-level metadata.

1.2. A Layered Model Over Existing Primitives

The identifiers defined by this document are not new inventions. The AGTP draft family already provides most of them. Agent-ID is defined in [AGTP] and derived from the Agent Genesis per [AGTP-LOG]. Owner-ID resolves to the principal recorded in the Agent Genesis. Session-ID is defined in [AGTP] and given session semantics by [AGTP-SESSION]. Task-ID is defined in [AGTP] for workflow correlation. The Attribution-Record envelope is defined in [AGTP] and already carries server identity, timestamp, request hash, and response status as a JWS-signed attestation on every response.

What this document adds is the connective layer:

* A small set of additional identifiers (Request-ID, Response-ID, Action-ID, Evaluation-ID, Decision-ID) that extend the Attribution-Record envelope to record the per-interaction and governance-evaluation events that base AGTP does not yet capture by themselves.

* The Audit-ID, defined as the cryptographic hash of the canonical extended Attribution-Record. The Audit-ID is the identifier of the audit record itself.

* The per-agent hash chain. Each Attribution-Record an agent emits carries the Audit-ID of the agent’s immediately preceding Attribution-Record. The chain is the agent’s accountability trail.

This document does not introduce a new transparency log. The identity-lifecycle transparency log defined in [AGTP-LOG] records Agent Genesis issuance and lifecycle events; this document operates against the per-response Attribution-Record envelope of [AGTP]. The two are complementary records of different concerns.

1.3. Identifier Layers

The identifier chain is organized into four layers, plus the Audit-ID that anchors them:

* *Persistent identifiers* are established at agent provisioning and remain stable for the agent’s operational lifetime: Agent-ID and Owner-ID.

* *Context identifiers* group multiple interactions under a shared operational context: Session-ID and Task-ID.

* *Per-interaction identifiers* are minted at the moment of each interaction and bind the request, the response, and the action taken: Request-ID, Response-ID, and Action-ID.

* *Governance identifiers* record the evaluation that authorized the action and the specific verdict produced: Evaluation-ID and Decision-ID.

The Audit-ID is the chain anchor that binds all of the above into a single verifiable artifact at each action.

The layer model is logical: it describes the role each identifier plays in the accountability chain. The cryptographic binding that makes the chain tamper-evident is established inside the extended Attribution-Record JWS, where every identifier present is signed together with the agent’s preceding Audit-ID. An identifier that appears in an Attribution-Record but cannot be linked to the records it claims to depend on (an Action-ID without its Evaluation-ID and Decision-ID, an Audit-ID whose previous_audit_id references no known predecessor) is a chain break and SHALL be treated as evidence of forgery or omission.

1.4. Relationship to the AGTP Draft Family

This document is the connective specification across the AGTP draft family. It does not replace any existing draft.

* [AGTP] provides the Attribution-Record envelope, the Agent-ID, Session-ID, and Task-ID headers, and the manifest signing infrastructure on which Attribution-Records are signed.

* [AGTP-TRUST] populates the Evaluation-ID record with the trust tier, verification path, trust score, and dimensional scoring defined there. Trust scoring is one set of dimensions within a governance evaluation.

* [AGTP-LOG] records identity-lifecycle events (Agent Genesis issuance, revocation, suspension, reinstatement, deprecation). Audit-IDs link to identity-lifecycle events when an agent's identity state changes, but Audit-IDs themselves are not committed to the AGTP-LOG transparency log.

* [AGTP-CERT] carries the Agent-ID in the subject-agent-id X.509 v3 extension. The Agent Certificate's TLS binding makes the wire identity verifiable at the transport layer.

* [AGTP-MERCHANT] already specifies that the Attribution-Record on a PURCHASE response carries merchant_id, merchant_fingerprint, and intent_assertion_jti fields. This document specifies how those merchant-specific fields coexist with the identifier chain fields in the same extended Attribution-Record.

* [AGTP-SESSION] introduces ESTABLISH semantics for Session-ID. This document treats Session-ID as a context identifier within the chain.

1.5. Requirements Language

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.

2. Terminology

This document uses the following terms.

Identifier Chain: The ordered set of identifiers defined by this document (Agent-ID, Owner-ID, Session-ID, Task-ID, Request-ID, Response-ID, Evaluation-ID, Action-ID, Decision-ID, Audit-ID) that together constitute a verifiable record of an agent action.

Persistent Identifier: An identifier whose value is established at agent provisioning and remains stable for the agent's operational lifetime. Agent-ID and Owner-ID are persistent identifiers.

Context Identifier: An identifier that groups multiple per-

interaction events under a shared operational context. Session-ID and Task-ID are context identifiers.

Per-Interaction Identifier: An identifier minted at the moment of a specific interaction. Request-ID, Response-ID, and Action-ID are per-interaction identifiers.

Governance Identifier: An identifier minted by a governance evaluation. Evaluation-ID and Decision-ID are governance identifiers.

Audit-ID: The cryptographic hash of the canonical extended Attribution-Record, computed as specified in Section 8.3. The Audit-ID is the identifier of a single audit record and the chain anchor that links the record to the predecessor record in the agent’s hash chain.

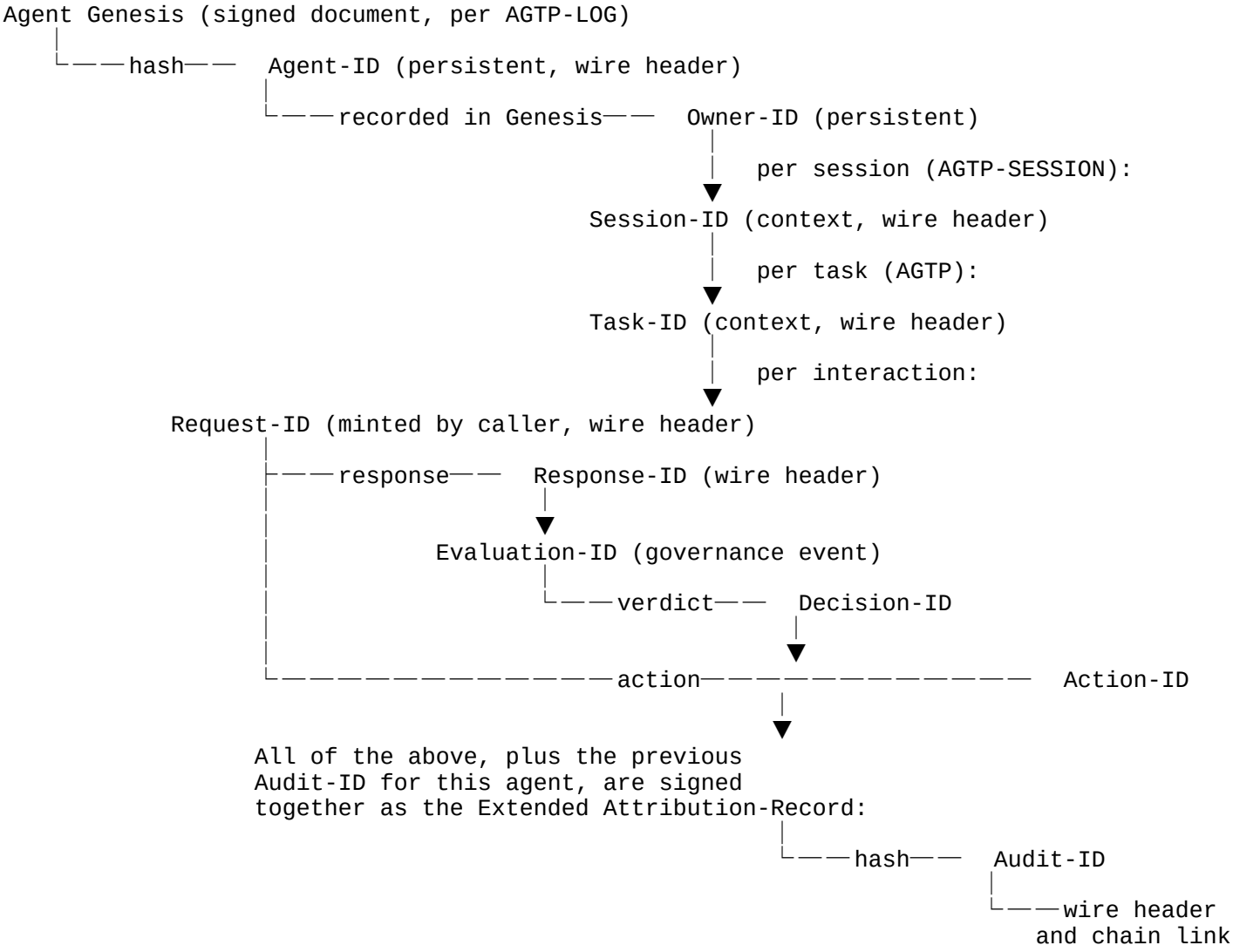
Extended Attribution-Record: The base AGTP Attribution-Record JWS envelope, augmented with the identifier-chain fields specified by this document. The extension is additive; existing Attribution-Record consumers remain conformant.

Chain Link: A cryptographic binding between two identifiers such that the successor identifier commits to the predecessor identifier. A chain link is established by including the predecessor identifier in the canonical input over which the successor identifier is computed or signed.

Chain Break: A condition in which an identifier in the chain cannot be linked to its predecessor. A chain break voids the audit value of the record and SHALL be treated as evidence of forgery or omission.

3. Identifier Stack Overview

The identifier chain is layered. Each identifier captures accountability for the moment it represents. The diagram below shows the relationship.



The chain reads top to bottom. Agent-ID is the foundational identifier from which the accountability chain originates. The Audit-ID is the per-action hash that closes the chain and links to the agent’s previous action.

4. Persistent Identifiers

4.1. Agent-ID

Agent-ID is the foundational identifier of the chain. It is the 256-bit cryptographic hash of the agent's Agent Genesis document, as defined by the Agent Identity Taxonomy in [AGTP-LOG]. Agent-ID is permanent for the operational lifetime of the agent; revocation produces a revoked state in the registry but does not retire or reissue the identifier.

Agent-ID MUST be:

* Derived from the Agent Genesis via the procedure specified in [AGTP-LOG]: agent_id = sha256(canonical_json(agent_genesis)).

* Rendered on the wire as 64 lowercase hexadecimal characters.

* Carried in the Agent-ID request header on every AGTP request, per [AGTP]. When the request carries Agent-ID, the response *SHOULD* echo the same value in the Agent-ID response header to correlate the response with the requesting agent. The responding host's identity is carried separately in the Server-ID response header per [AGTP]; Server-ID names the host that processed the request and is not an agent identifier. When the audit consumer needs the canonical Agent-ID of the agent that produced the response, it is recorded in the agent_id field of the Attribution-Record payload.

* Resolvable to an Agent Identity Document by any AGTP-aware infrastructure component, via the resolution mechanics specified in [AGTP].

Agent-ID identifies who the actor is throughout the agent's operational lifetime. Every subsequent identifier in the chain links back to Agent-ID. An action whose Agent-ID cannot be resolved is an anonymous action and SHALL be refused by AGTP-aware infrastructure.

4.2. Owner-ID

Owner-ID is the identifier of the human or organizational principal accountable for the agent. Owner-ID is bound to Agent-ID at provisioning and recorded in the Agent Genesis document. The binding is asserted by the governance platform that issued the Agent Genesis and is verifiable through the signature on the Genesis document.

Owner-ID MUST be:

* Present in the Agent Genesis document as the owner_id field.

* Carried in the Owner-ID request header on every AGTP request, alongside the Agent-ID header.

* Resolvable to an Owner Identity Document that records the legal identity of the principal (a natural person, a registered organization, or a legal entity recognized by the governance platform's jurisdiction).

* Stable for the operational lifetime of the agent. Ownership transfer requires reissuance of the Agent Genesis and produces a new Agent-ID.

Owner-ID MAY use any identifier scheme the governance platform recognizes: a national identity number where lawful, a registered organization number, a DID, or a platform-issued identifier. The identifier scheme MUST be declared in the Owner Identity Document under the id_scheme field.

An Agent-ID without a resolvable Owner-ID lacks an accountability terminus and SHALL be treated as a Tier 3 (Experimental) agent per [AGTP-TRUST], regardless of any other trust signals it carries.

4.3. Why Both Identifiers Are Required

Agent-ID answers the question "what acted." Owner-ID answers the question "who is responsible." A regulatory examination that asks "who is liable for this action" requires both. An incident response that asks "which human do we contact" requires Owner-ID. A counterparty that asks "is this agent who it claims to be" requires Agent-ID.

The two identifiers MUST appear together on every request. A request carrying Agent-ID without Owner-ID is a malformed AGTP request and SHALL be refused with status 400 Bad Request.

5. Context Identifiers

5.1. Session-ID

Session-ID identifies a logical context shared between two or more AGTP participants under which multiple method invocations are correlated. Session-ID is defined as a request header by [AGTP] and given establishment semantics by [AGTP-SESSION].

For the purposes of the identifier chain:

* Session-ID MAY be absent. Many agent interactions are single-method exchanges that have no session context.

* When present, Session-ID MUST be carried on every request and response within the session, per [AGTP-SESSION].

* Session-ID MUST be included in the extended Attribution-Record when present on the request that triggered the response.

A session collects multiple Request-ID, Response-ID, and Action-ID triples under one operational context. The session is not itself an action; it is the context within which actions occur.

5.2. Task-ID

Task-ID identifies a specific task or operation that spans multiple requests, typically within a workflow that involves multiple agents or services. Task-ID is defined as a request and response header by [AGTP].

For the purposes of the identifier chain:

* Task-ID MAY be absent. Many actions are not part of a multi-request task.

* When present, Task-ID MUST be carried on every request and response that participates in the task, per [AGTP].

* Task-ID MUST be included in the extended Attribution-Record when present on the request that triggered the response.

Task-ID and Session-ID are independent. A long session may contain multiple discrete tasks. A single task may span multiple sessions if the workflow requires it. The two identifiers MAY both be present on the same request.

6. Per-Interaction Identifiers

6.1. Request-ID

Request-ID captures the initiation of a specific interaction. When an agent issues a request to a tool, an API, another agent, or a service, the caller MUST mint a Request-ID and include it on the outbound request.

Request-ID MUST be:

* A UUIDv7 per [RFC9562] or a ULID rendered as 26 base32 characters, selected at the caller’s discretion. Both schemes provide monotonic ordering and high collision resistance.

* Minted by the calling agent, not by the server. Server-minted identifiers break the causal chain because the caller cannot prove it issued the request before the server received it.

* Carried in the Request-ID request header.

* Unique within the caller's identifier space for the lifetime of the calling agent. Collisions across agents are tolerated; collisions within an agent's own request space are a conformance failure.

The Request-ID enables tracing an action back to its origin. An audit trail that records an action but cannot identify the request that initiated it cannot survive examination. Without Request-ID, the record shows that something happened but not what initiated it.

6.2. Response-ID

Response-ID links the response to the request that generated it. The responding agent MUST mint a Response-ID and include it on the outbound response.

Response-ID MUST be:

* A UUIDv7 or ULID, with the same construction rules as Request-ID.

* Minted by the responding agent.

* Carried in the Response-ID response header.

* Paired with the corresponding Request-ID by echoing the request's Request-ID header on the response, following the same echo convention [AGTP] uses for Task-ID. The responding agent SHOULD echo Request-ID unchanged in the response.

Without Response-ID, the audit trail can show that a request was made and that an action followed, but cannot prove that the specific response to the specific request was what drove the action. The linkage between what the agent received and what it did next requires both identifiers to be present.

6.3. Action-ID

Action-ID captures the specific action taken. An agent may take multiple actions within a session or task; each action MUST have its own Action-ID to be separately traceable.

Action-ID MUST be:

* Minted by the agent at the moment of action.

* A UUIDv7 or ULID, with the same construction rules as Request-ID.

* Carried as the action_id field in the extended Attribution-Record. Action-ID MAY also be carried on the wire as the Action-ID response header for client convenience.

* Co-located with the Request-ID that initiated the action and the Response-ID that informed it inside the same extended Attribution-Record JWS. The JWS signature binds all three identifiers together as a single signed unit.

Action-ID MUST be present in the extended Attribution-Record for any method that produces state change, including (but not limited to) EXECUTE, DELEGATE, ESCALATE, CONFIRM, SUSPEND, PURCHASE, and the identity-lifecycle methods that trigger AGTP-LOG events (ACTIVATE, REVOKE, REINSTATE, DEPRECATE per [AGTP-LOG]). Action-ID MAY be absent for cognitive methods (QUERY, DISCOVER, DESCRIBE, SUMMARIZE, PLAN, PROPOSE) that return information without recording an action in the accountability sense; the Attribution-Record is still emitted but the action_id field is omitted.

7. Governance Identifiers

7.1. Evaluation-ID

Evaluation-ID is the identifier for the governance evaluation applied before an action proceeded. This is the record of the evaluation event, distinct from the verdict that the evaluation produced.

Evaluation-ID MUST be:

* Minted by the governance system at the moment the evaluation begins.

* A UUIDv7 or ULID.

* Carried as the evaluation_id field in the extended Attribution-Record.

* Resolvable to an Evaluation Record per Section 7.1.1 that captures the inputs evaluated, the dimensional scores computed, the confidence level, and the verdict produced.

7.1.1. Evaluation Record

The Evaluation Record is held by the governance platform that performed the evaluation. The Evaluation Record SHALL contain:

* evaluation_id (this identifier).
* agent_id (the agent being evaluated).
* owner_id (the owner accountable for the agent).
* request_id (the request that triggered the evaluation).
* contract_id (the identifier of the governance contract or policy in effect at the time of evaluation; the contract scheme is operator-defined and tracked as an Open Item in this revision).
* inputs (a canonical representation of the inputs evaluated, with PII redacted per the governance platform's policy).
* dimension_scores (a map of dimension names to scores, per [AGTP-TRUST]).
* confidence (a scalar in the range 0.0 to 1.0 expressing the evaluator's confidence in its dimensional measurements).
* timestamp_start (RFC 3339 timestamp of evaluation initiation).
* timestamp_end (RFC 3339 timestamp of evaluation completion).

The Evaluation Record is signed by the governance platform's key and stored in the governance platform's evaluation store. The Attribution-Record references the Evaluation Record by evaluation_id; the full Evaluation Record is retrievable out-of-band via the governance platform's evaluation lookup endpoint.

An audit trail without Evaluation-IDs records what happened but cannot prove that a governance evaluation occurred. Under the EU AI Act and similar regimes, the absence of an Evaluation-ID for a high-risk action is itself a compliance failure.

7.2. Decision-ID

Decision-ID captures the governance verdict specifically. The Evaluation-ID covers the full evaluation process; the Decision-ID is the output of that process.

Decision-ID MUST be:

* Minted by the governance system at the moment the verdict is produced.

* A UUIDv7 or ULID.

* Carried as the decision_id field in the extended Attribution-Record.

* Linked to the Evaluation-ID that produced it via the evaluation_id field in the Decision Record.

7.2.1. Decision Record

The Decision Record is held by the governance platform that produced the verdict. The Decision Record SHALL contain:

* decision_id (this identifier).

* evaluation_id (the evaluation that produced this decision).

* verdict (one of permit, deny, permit-with-conditions, defer).

* reasoning (a structured record of the reasoning that produced the verdict, in a format the governance platform defines).

* conditions (when verdict is permit-with-conditions, the conditions under which the permit applies).

* timestamp (RFC 3339 timestamp of verdict issuance).

* valid_until (RFC 3339 timestamp after which the decision MUST be re-evaluated; absent for one-shot decisions).

The Decision Record is signed by the governance platform’s key and stored in the governance platform’s decision store. As with Evaluation Records, the Attribution-Record references the Decision Record by decision_id; the full record is retrievable out-of-band.

The Decision-ID is what makes the verdict independently auditable. A regulator or attorney requesting the governance decision for a specific action MUST be able to retrieve the Decision-ID and read the full reasoning record that produced it.

7.2.2. Separation of Evaluation and Decision

Evaluation-ID and Decision-ID are separate identifiers because the two concepts are auditable separately. A regulator may ask "show me the inputs to the evaluation" (Evaluation Record) or "show me the verdict and the reasoning" (Decision Record). The governance platform MAY produce multiple Decision Records from a single Evaluation Record (for example, when a deferred decision is later resolved); each Decision Record references the same Evaluation-ID via the evaluation_id field.

8. Audit Identifier and the Attribution-Record Chain

8.1. Audit-ID

Audit-ID is the cryptographic hash of the canonical extended Attribution-Record. It is the identifier of a single audit record and the chain anchor that links the record to the agent’s preceding audit record.

Audit-ID MUST be:

* A 256-bit value rendered as 64 lowercase hexadecimal characters.

* Computed as the SHA-256 hash of the canonical extended Attribution-Record JWS, per Section 8.3.

* Carried on the wire as the Audit-ID response header on every response that carries an Attribution-Record.

* Included as the previous_audit_id field in the agent’s next emitted Attribution-Record, completing the chain link.

8.2. Extension to the Base Attribution-Record

The base Attribution-Record envelope defined by [AGTP] is a JWS-signed token carrying server identity, response timestamp, request hash, and response status. This document extends the Attribution-Record payload with the identifier-chain fields. The extension is additive: existing Attribution-Record consumers remain conformant, and the JWS signature continues to cover the full extended payload.

The extended Attribution-Record payload contains the following identifier-chain fields:

Field	Type	Required	Description
agent_id	string	*MUST*	Acting agent's 64-hex Agent-ID
owner_id	string	*MUST*	Owner identifier per Section 4.2
session_id	string	conditional	Present when the request carried Session-ID
task_id	string	conditional	Present when the request carried Task-ID
request_id	string	*MUST*	Request-ID of the request being responded to
response_id	string	*MUST*	Response-ID of this response
action_id	string	conditional	Present for action-recording methods
evaluation_id	string	conditional	Present when governance evaluation occurred
decision_id	string	conditional	Present when governance decision was rendered
previous_audit_id	string	*MUST*	The agent's preceding Audit-ID, or 64 zeros for the first record
audit_record_version	string	*MUST*	This document's version: "1"
requested_method	string	conditional	The method as it arrived on the wire when alias resolution per [AGTP-API]

			translated it to a different canonical method before dispatch. Present only when an alias was resolved; absent when the request method was already canonical. Allows audit consumers to distinguish "BOOK arrived under that name" from "GET arrived and was aliased to FETCH."
acting_principal_id	string	conditional	Identifier of the human or service principal on whose behalf the agent acted, lifted from a validated external-IdP credential per the OAuth composition section in [AGTP]. Set when the server is configured to validate external credentials and the validator succeeded; the specific claim lifted is operator-configured (typically sub for OIDC). Absent when the request carried no external credential, when the server was not configured to validate one, or when validation produced no lifted claim. Distinct from owner_id (which identifies the agent’s principal as

			recorded on the Agent Genesis) and from the Agent-ID: this field identifies the principal who delegated this specific request to the agent.
synthesis_id	string	conditional	Present when the action was dispatched under a synthesized contract (RCNS-spawned or PROPOSE-spawned). Opaque identifier of the contract per [AGTP-API].
contract_hash	string	conditional	Present when the action was dispatched under a synthesized contract. SHA-256 of the canonical contract definition, 64 lowercase hex characters. Stable across all invocations of the same contract.
negotiation_origin	string	conditional	Present when the action was dispatched under a synthesized contract. One of rcns-gate, propose, or pre-registered.

Table 1: Identifier-Chain Fields in the Extended Attribution-Record

These fields coexist with merchant-specific fields specified by [AGTP-MERCHANT] (merchant_id, merchant_fingerprint, intent_assertion_jti), with the base Attribution-Record fields from [AGTP], and with any other companion-draft extensions to the Attribution-Record payload.

8.3. Audit-ID Construction

The Audit-ID is computed as follows:

1. The responding agent assembles the extended Attribution-Record payload as a JSON object containing the base AGTP fields, the identifier-chain fields from Table 1, and any other companion-draft extensions applicable to the response.

2. The payload is signed using the responding agent’s manifest signing key per [AGTP]. The result is the JWS that the Attribution-Record response header carries.

3. The Audit-ID is computed as audit_id = sha256(jws_compact_serialization), where jws_compact_serialization is the full JWS compact form (header.payload.signature, base64url-encoded per [RFC7515]).

4. The responding agent stores the Audit-ID in its local audit store, indexed by agent_id and ordered by emission time. The same Audit-ID becomes the previous_audit_id value in the agent’s next emitted Attribution-Record.

The hash covers the full signed JWS, not the JSON payload. This ensures that any modification of the signature, the protected header, or the payload invalidates the Audit-ID.

8.3.1. Unsigned Fallback (alg: none)

A responding agent that has not yet provisioned a manifest signing key MAY emit Attribution-Records with alg: none in the JWS protected header and an empty signature octet, per [RFC7515] Section 4.1.1. The Audit-ID is still computed as sha256(jws_compact_serialization) over the full unsigned compact form, and the per-agent chain is maintained: the next record carries this Audit-ID in its previous_audit_id field exactly as it would for a signed record.

The unsigned fallback preserves wire format and chain construction during development and transition. It carries no anti-forgery claim. Verifiers and audit-trail consumers *MUST* treat alg: none records as unverified per the Attribution Forgery mitigation in [AGTP]. An agent that emits alg: none records once and signed records later

produces a chain in which the unsigned prefix is hash-linked to the signed continuation; verifiers can audit the signed portion without re-evaluating the unsigned prefix, and the chain itself remains intact.

Production deployments *MUST* configure a manifest signing key. The unsigned fallback is *NOT* a long-term mode.

8.4. The Per-Agent Hash Chain

Each agent maintains a hash chain over its own emitted Attribution-Records. The chain has the following properties:

* The chain head is the first Attribution-Record the agent ever emits. Its previous_audit_id is 64 hex zeros.

* Each subsequent Attribution-Record carries the Audit-ID of the immediately preceding Attribution-Record this agent emitted in its previous_audit_id field.

* Tampering with any record in the chain invalidates the previous_audit_id chain link for every record that follows it, and invalidates the JWS signature on the tampered record.

The chain is scoped per agent. An agent’s chain is independent of any other agent’s chain. A verifier reconstructing one agent’s actions follows only that agent’s chain.

The agent is responsible for maintaining chain integrity. An agent that emits an Attribution-Record without setting previous_audit_id correctly produces a chain break and SHALL have its Tier 1 verification status revoked per [AGTP-TRUST].

8.5. Cross-Agent Chain Linkage

A single action sometimes references actions taken by other agents. A delegation chain in which Agent C delegates to Agent A, which delegates to Agent B, produces three distinct per-agent chains. Each agent maintains its own chain; cross-agent references link them where required.

When the action recorded in an Attribution-Record depends on or continues an action recorded by a different agent, the extended Attribution-Record MUST include a prior_actions array. Each entry is a tuple of the form:

```
{
  "agent_id": "<64 hex>",
  "audit_id": "<64 hex>",
  "agent_uri": "<AGTP URI>"
}
```

The agent_uri field is OPTIONAL but RECOMMENDED. When present, it carries the AGTP URI at which the referenced agent's audit store was addressable at the time the recording agent emitted the Attribution-Record (typically the URI the recording agent just invoked to obtain the upstream Audit-ID). A verifier that walks prior_actions SHOULD prefer the inline agent_uri over any locally-cached agent location, since the recording agent authored the URI with first-hand knowledge of where the upstream agent lived at the time of the interaction; locally-cached locations may have drifted out of date.

A verifier traversing the chain MAY use prior_actions entries to walk into other agents' chains. Resolution of a prior_actions entry uses the standard Agent-ID resolution path when agent_uri is absent: Agent-ID resolves to the Agent Genesis per [AGTP-LOG], the Genesis names the governance platform that issued it, and the Audit-ID is retrievable from that platform's audit infrastructure for that agent. When agent_uri is present, the verifier MAY use it directly and skip the resolution step. The standardized retrieval interface, regardless of how the verifier located the agent's audit store, is the INSPECT method specified in [AGTP]: a verifier invokes INSPECT target=audit audit_id={audit_id} on the referenced agent's server.

Multiple entries with the same (agent_id, audit_id) pair MAY appear in the array when the recording agent's action depended on the same upstream action through multiple intermediate agents; verifiers MUST treat duplicate entries as references to a single upstream node (diamond convergence) and MUST NOT double-count traversal steps.

The combination of per-agent chains and prior_actions references produces a verifiable global graph of action provenance. A regulator examining an incident MAY traverse the graph from any terminal Audit-ID back to the originating Agent Genesis through as many agents as the action involved.

8.6. Field Population Rules

The following fields MAY be absent under specific conditions:

* evaluation_id and decision_id MAY be absent for actions taken under standing authorization (a previously issued permit-with-conditions decision still within its valid_until window). In that case the extended Attribution-Record MUST include a standing_authorization_decision_id field pointing to the Decision-ID that authorized the action.

* action_id MAY be absent for cognitive methods that do not record actions, as specified in Section 6.3.

* session_id MAY be absent when the request was not part of a session.

* task_id MAY be absent when the request was not part of a multi-request task.

* All other fields MUST be present. A field that MUST be present but is absent constitutes a chain break.

8.7. Verification Procedure

A verifier reconstructs the chain by:

1. Retrieving the terminal Audit-ID from the agent’s audit store or from a counterparty’s record of the Attribution-Record it received.

2. Fetching the full Attribution-Record JWS and verifying its hash matches the Audit-ID.

3. Verifying the JWS signature against the responding agent’s manifest signing key, per [AGTP].

4. Following previous_audit_id backwards through the agent’s chain, verifying each hash and each signature, until the chain head is reached.

5. When the Attribution-Record carries a prior_actions array, resolving each {agent_id, audit_id} tuple via the standard Agent-ID resolution path and verifying the referenced Attribution-Record by the same procedure recursively.

6. For each Attribution-Record, retrieving the referenced Evaluation Record and Decision Record from the governance platform and verifying each is signed by the appropriate governance platform key.

7. Confirming the Agent-ID resolves to a valid Agent Genesis per [AGTP-LOG], that the Owner-ID matches the Genesis's owner_id field, and that the agent's identity state at the time of each Attribution-Record was consistent with the action recorded (the agent was not revoked or suspended at the time of action).

8. Confirming all timestamps are monotonically consistent (Request-ID timestamp precedes Response-ID timestamp, which precedes or is concurrent with Action-ID timestamp, with Evaluation and Decision timestamps before Action-ID).

A verification failure at any step is a chain break and invalidates the audit value of the record.

9. Wire Format

The identifier chain materializes on the wire as a small set of new request and response headers and as extensions to the existing Attribution-Record envelope. Headers defined by other AGTP drafts (Agent-ID, Owner-ID, Session-ID, Task-ID, Server-ID, Attribution-Record) are referenced by this document and not redefined here.

9.1. New Headers Defined by This Document

Header	Carried On	Required	Value
Owner-ID	request	*MUST*	Owner identifier per Section 4.2
Request-ID	request	*MUST*	UUIDv7 or ULID per Section 6.1
Request-ID	response	*SHOULD*	Echo of the originating request's Request-ID
Response-ID	response	*MUST*	UUIDv7 or ULID per Section 6.2
Audit-ID	response	*MUST* when Attribution-Record present	64 lowercase hex per Section 8.1
Action-ID	response	*MAY*	UUIDv7 or ULID; convenience echo

			of the action_id field in the Attribution-Record payload

Table 2: New Headers Defined by AGTP-IDENTIFIERS

Action-ID, Evaluation-ID, and Decision-ID are primarily carried as fields inside the extended Attribution-Record payload, not as standalone wire headers. The Action-ID MAY appear as a convenience header on the response for clients that need to reference it without parsing the JWS.

9.2. ABNF Grammar

The header values follow this grammar:

```
owner-id       = 1*256(ALPHA / DIGIT / "-" / "_" / ":" / ".")
request-id     = uuidv7 / ulid
response-id    = uuidv7 / ulid
audit-id       = 64HEXDIGLOWER
action-id      = uuidv7 / ulid

uuidv7         = 8HEXDIGLOWER "-" 4HEXDIGLOWER "-" "7" 3HEXDIGLOWER
                 "-" 4HEXDIGLOWER "-" 12HEXDIGLOWER
ulid           = 26BASE32

HEXDIGLOWER    = DIGIT / "a" / "b" / "c" / "d" / "e" / "f"
BASE32         = ALPHA / DIGIT
```

A header value that does not conform to this grammar SHALL be treated as malformed and the request or response SHALL be refused with status 400 Bad Request.

9.3. Echo Rules

The responding agent SHOULD echo the request's Request-ID unchanged in the response. This follows the echo convention [AGTP] uses for Task-ID and Agent-ID, and avoids introducing a separate correlation header. The host that processed the request is named in Server-ID per [AGTP]; Server-ID is a host identifier and is not an agent identity. The canonical Agent-ID of the agent that produced the response is recorded in the agent_id field of the Attribution-Record payload. The responder's per-interaction identifier is carried in Response-ID.

A response that omits Request-ID when responding to a request that carried Request-ID produces an audit chain that cannot be reconstructed without out-of-band correlation, and SHOULD be treated as a chain break by infrastructure relying on the identifier chain for accountability.

10. Identifier Resolution and Lookup

10.1. Resolution Targets

Each identifier resolves to a specific record. The resolution target and store are summarized below.

Identifier	Resolves To	Store
Agent-ID	Agent Identity Document	Governance platform registry; AGTP-LOG for lifecycle events
Owner-ID	Owner Identity Document	Governance platform
Session-ID	Session Context	Session participants and governance platform per [AGTP-SESSION]
Task-ID	Task Context	Operator-defined
Request-ID	(no separate record; appears in Attribution-Record)	Caller’s outbound log; responder’s Attribution-Record
Response-ID	(no separate record; appears in Attribution-Record)	Responder’s Attribution-Record
Action-ID	Extended Attribution-Record	Responding agent’s audit store
Evaluation-ID	Evaluation Record	Governance platform’s evaluation store
Decision-ID	Decision Record	Governance platform’s decision store
Audit-ID	Extended Attribution-Record JWS	Responding agent’s audit store; counterparty’s received-record store

Table 3: Identifier Resolution Targets

10.2. Lookup Methods

The default lookup method is the responding agent’s audit store, which MUST be addressable by Audit-ID. The standardized AGTP lookup interface is the INSPECT method specified in [AGTP]: a relying party invokes INSPECT target=audit audit_id={hex} on the agent’s server and receives the JWS Compact serialization of the stored Attribution-Record. The relying party *MUST* recompute sha256(jws) and confirm

it equals the requested audit_id before trusting the parsed payload. INSPECT also supports target=chain_head (returns the most recently emitted Audit-ID for a named Agent-ID, useful for chain walkers needing a starting point) and target=lifecycle (returns recent lifecycle log entries for a named Agent-ID).

Implementations *MAY* additionally expose a REST endpoint over HTTPS or rely on out-of-band retrieval; the INSPECT method is the protocol-level interface and the recommended default.

Evaluation Records and Decision Records are stored by the governance platform that produced them. Lookup is via the governance platform's evaluation and decision endpoints (out of scope for this document; defined by the governance platform).

Agent-ID and Owner-ID resolve via the mechanisms defined in [AGTP]. Identity-lifecycle events for Agent-ID are stored in the AGTP-LOG transparency log per [AGTP-LOG]; the lookup is via the AGTP-LOG protocol.

10.3. Caching

Persistent identifiers (Agent-ID, Owner-ID) MAY be cached indefinitely by infrastructure components, refreshing on revocation signals from the AGTP-LOG transparency log.

Per-interaction and governance identifiers SHOULD be cached only for the duration of the operation that references them. Caching beyond the immediate operation is an operator policy concern and is out of scope for this document.

11. Security Considerations

11.1. Identifier Forgery

The primary threat is forgery of any identifier in the chain. The chain is resistant to forgery because:

* Agent-ID is the hash of a signed Agent Genesis document. Forging Agent-ID requires forging the Genesis signature.

* Owner-ID is bound to Agent-ID via the Genesis signature. Forging Owner-ID requires forging Agent-ID.

* Per-interaction identifiers (Request-ID, Response-ID, Action-ID) are committed inside the JWS-signed extended Attribution-Record. Forging an after-the-fact identifier requires forging the JWS signature.

* Governance identifiers (Evaluation-ID, Decision-ID) are referenced inside the JWS-signed Attribution-Record and produced by the governance platform's signed records. Forging requires compromising the governance platform's key or forging the JWS.

* Audit-ID is the SHA-256 hash of the canonical JWS. Forging Audit-ID requires producing a colliding JWS, which is computationally infeasible.

A verifier that retrieves an identifier from a source other than the responding agent's audit store or a signed Attribution-Record SHALL treat the identifier as unverified until the originating record is retrieved and the signature confirmed.

11.2. Chain Breaks

A chain break is the condition in which an identifier in the chain cannot be linked to its predecessor. Chain breaks are caused by:

* Omission (the predecessor identifier was never minted).

* Tampering (the predecessor identifier was modified after issuance).

* Substitution (a different identifier was inserted in place of the original).

All three conditions MUST be treated as evidence of forgery or omission. The audit record SHALL NOT be relied upon for regulatory examination or counterparty dispute.

11.3. Replay

An attacker that captures an extended Attribution-Record MAY attempt to replay it as if it were a new action. The chain resists replay because:

* The Attribution-Record carries a timestamp inside the signed payload. Replay produces a stale timestamp that verifiers can reject.

* The previous_audit_id field commits to a specific predecessor. A replayed record carries the same previous_audit_id as its original, which cannot match the current chain head when the chain has progressed.

* The Audit-ID is deterministic over the JWS. A replayed Attribution-Record produces the same Audit-ID as its original, which the responding agent's audit store rejects as a duplicate.

The responding agent MUST enforce strict append-only semantics on its audit store. An audit store that permits insertion or modification of past entries undermines the replay resistance of the chain.

11.4. Agent Compromise

If the responding agent's manifest signing key is compromised, an attacker can produce valid-looking Attribution-Records under that agent's identity. The chain provides no protection against this case; the threat is mitigated at the [AGTP-LOG] layer by issuing a agent-genesis-revoked event that retires the compromised agent.

A verifier that finds an agent-genesis-revoked event for an Agent-ID SHOULD treat all Attribution-Records emitted under that Agent-ID after the revocation timestamp as suspect. Attribution-Records emitted before revocation remain valid; they were emitted by the agent in good standing at the time.

11.5. Privacy

Identifiers themselves carry no payload data. However, the records they resolve to MAY contain personally identifiable information (particularly in Evaluation Records, where inputs may include user-submitted content).

The governance platform that issues an identifier is responsible for the privacy of the record it resolves to. Specifically:

* Owner-ID records MAY identify a natural person and MUST be treated as personal data under applicable law (GDPR, CCPA, and similar regimes).

* Evaluation Records' inputs field SHOULD be redacted to remove PII before the record is committed to the evaluation store. The governance platform MAY retain the unredacted form under separate access controls.

* Request-IDs and Response-IDs themselves are not personal data; the payloads they reference MAY be.

A verifier with regulatory authority MAY request the unredacted form via the governance platform's out-of-band access procedure. A verifier without such authority is bound by the redacted form.

11.6. Identifier Reuse

An identifier issued under one Agent-ID SHALL NOT be reused under a different Agent-ID. The collision-resistant identifier schemes (UUIDv7, ULID, SHA-256) make this statistically improbable. Governance platforms MUST enforce this constraint during identifier issuance.

12. Privacy Considerations

The identifier chain produces a complete record of an agent's actions, indexed by Agent-ID and Owner-ID. This is by design: the chain is the audit record. However, the design has privacy consequences that implementers MUST consider.

12.1. Linkability

The chain links every action an agent takes to a single Agent-ID, which in turn links to a single Owner-ID. An adversary with access to a responding agent's audit store or to multiple Attribution-Records emitted by the same agent can construct a behavioral profile.

Mitigations:

* Sensitive Owner-IDs (those identifying natural persons) MAY be pseudonymized at the governance platform. The platform retains the binding between pseudonym and natural person under separate access controls.

* An owner that requires unlinkability across agent activities MAY provision multiple Agent-IDs under separate Owner-IDs. The governance platform MAY support this directly via the Owner Identity Document.

12.2. Right to Erasure

GDPR Article 17 and similar provisions establish a right to erasure of personal data. The append-only nature of the agent's audit store creates tension with this right.

A governance platform that operates in a jurisdiction with a right to erasure MUST implement one of:

* A redaction mechanism that nulls the personal data fields in audit entries without removing the entries themselves, preserving the chain.

* A revocation mechanism that marks the Owner-ID as revoked and prevents future linkage, preserving the chain for past entries under restricted access.

The specific mechanism is out of scope for this document.

13. Regulatory Mapping

The identifier chain is designed to satisfy the record-keeping requirements of major AI regulatory regimes. This section is informative.

13.1. EU AI Act

The EU AI Act [EU-AI-ACT] requires that high-risk AI systems maintain records sufficient to reconstruct decisions. The chain satisfies this through:

* Agent-ID and Owner-ID identify the actor and the accountable principal.

* Evaluation-ID and Decision-ID record the governance evaluation and verdict required by Article 12 (logging) and Article 14 (human oversight).

* Audit-ID provides the tamper-evident binding required by Article 20 (record-keeping) and Article 26 (deployer obligations).

A high-risk AI system that maintains the AGTP identifier chain maintains the records the EU AI Act requires.

13.2. SOC 2 and ISO 42001

SOC 2 Type II audits require evidence that controls operate effectively over time. ISO 42001 requires a management system for AI with comparable evidentiary requirements. The chain provides this evidence through cryptographically verifiable records of:

* Who acted (Agent-ID, Owner-ID).

* What authorized the action (Evaluation-ID, Decision-ID).

* What action was taken (Action-ID).

* When and in what order (timestamps and previous_audit_id).

14. IANA Considerations

This document requests the following IANA actions.

14.1. AGTP Header Field Registry Additions

Registration of the following entries in the AGTP Header Field Registry (established by [AGTP]):

Header	Reference	Carried On	Required
Owner-ID	Section 4.2	request	*MUST*
Request-ID	Section 6.1	request	*MUST*
Request-ID	Section 6.1	response	*SHOULD* (echo)
Response-ID	Section 6.2	response	*MUST*
Audit-ID	Section 8.1	response	*MUST* with Attribution-Record
Action-ID	Section 6.3	response	*MAY*

Table 4

14.2. AGTP Attribution-Record Field Registry

Establishment of an IANA registry: AGTP Attribution-Record Fields, with initial registrations for the identifier-chain fields specified in this document (Table 1), the merchant-identity fields specified in [AGTP-MERCHANT], and the base fields specified in [AGTP].

Registration Procedure Specification Required

14.3. AGTP Verdict Registry

Establishment of an IANA registry: AGTP Decision Verdicts, with initial registrations for permit, deny, permit-with-conditions, and defer.

Registration Procedure Specification Required

14.4. AGTP Identifier Scheme Registry

Establishment of an IANA registry: AGTP Identifier Schemes, with initial registrations for uuidv7, ulid, sha256, and the id_scheme values recognized for Owner-ID.

Registration Procedure Specification Required

15. Open Items

The following items are out of scope for this revision and are anticipated in future revisions:

* Contract ID specification. Evaluation Records reference a contract_id (the governance contract or policy in effect at the time of evaluation). The contract ID scheme is referenced but not defined here.

* Concrete redaction format for inputs in Evaluation Records.

* The Owner Identity Document format. This document specifies what Owner-ID resolves to but does not define the document structure.

* Cross-agent audit chain federation protocol. The wire format for prior_actions references in the Attribution-Record payload is specified; the federation protocol by which governance platforms reconcile cross-agent chains and serve remote audit lookups remains to be defined.

* Standing authorization expiry and re-evaluation triggers.

* Identifier scheme migration. The current document permits UUIDv7 and ULID for per-interaction and governance identifiers. A future revision may add or constrain schemes.

* Reconciliation of the retired Principal-ID header in [AGTP] with the Owner-ID header introduced by this document. A future revision of [AGTP] is expected to align on Owner-ID as the wire header carrying owner identity.

16. Changes from v00

Version 01 is a drift-cleanup revision. The identifier stack is unchanged; corrections align spec wording with the deployed Agent-ID echo behavior documented in [AGTP] v08.

16.1. Substantive Changes

The following substantive changes were made:

1. *Agent-ID echo on responses corrected.* The Agent-ID section (Section 4.1) previously stated that responses carry the responding agent's identity in the Server-ID header rather than echoing the requester's Agent-ID. This was wrong on two counts: Server-ID is a host identifier and is not an agent identifier, and responses do echo the request's Agent-ID per [AGTP]. The corrected text states that responses *SHOULD* echo Agent-ID, that Server-ID names the host that processed the request, and that the canonical Agent-ID of the agent that produced the response is recorded in the agent_id field of the Attribution-Record payload.

2. *Echo Rules section corrected.* The same Server-ID-as- responder-identity confusion appeared in the Echo Rules section under Request-ID and has been corrected with parallel wording. The Request-ID echo guidance is unchanged in substance; only the surrounding identity framing has been fixed.

3. *alg: none fallback documented.* The Audit-ID Construction section (Section 8.3) now includes an Unsigned Fallback subsection covering the case where a responding agent has not provisioned a manifest signing key. The fallback emits a JWS with alg: none per [RFC7515] Section 4.1.1; the Audit-ID is still computed and the chain is still maintained. The fallback carries no anti-forgery claim and is bounded to development and transition deployments. Verifiers *MUST* treat alg: none records as unverified. This documents behavior already deployed in v07-conformant implementations and matches the corresponding tightening in [AGTP] v08.

4. *INSPECT named as the standardized Audit-ID lookup interface.* The Lookup Methods section is updated to specify the INSPECT method defined in [AGTP] v08 as the standardized AGTP lookup interface for Audit-IDs. A relying party that holds an Audit-ID invokes INSPECT target=audit audit_id={hex} on the responding agent's server and receives the JWS Compact serialization of the stored Attribution-Record. INSPECT additionally supports target=chain_head and target=lifecycle for chain walkers and lifecycle auditors. The earlier text that left lookup operator-defined and out of scope is superseded; HTTPS REST endpoints and out-of-band retrieval remain permitted as additional surfaces. The AGTP-MERCHANT informative seriesinfo is also updated to v02 to track the unification of merchant identity onto the Agent Identity Document.

5. *prior_actions entries gain optional agent_uri.* The schema for each prior_actions array entry is extended with an OPTIONAL but RECOMMENDED agent_uri field carrying the AGTP URI at which the referenced agent's audit store was addressable at the time the recording agent emitted the Attribution-Record. Verifiers *SHOULD* prefer the inline agent_uri over locally-cached agent locations on grounds that the recording agent authored the URI with first-hand knowledge of the upstream agent's location. The retrieval interface, regardless of how the agent is located, is INSPECT per entry #4. Cross-agent walkers MAY encounter diamond convergence (multiple prior_actions entries referencing the same (agent_id, audit_id) pair); verifiers *MUST* treat duplicates as references to a single upstream node and *MUST NOT* double-count traversal steps. This documents the cross-agent BFS walker shape that deployed chain inspectors have shipped.

6. *Extended Attribution-Record gains four conditional companion-draft fields.* The Identifier-Chain Fields table is extended with four new entries: requested_method (set when method-alias resolution per [AGTP-API] translated the wire method to a canonical method before dispatch); synthesis_id, contract_hash, and negotiation_origin (set when the action was dispatched under a synthesized contract — RCNS-spawned or PROPOSE-spawned — per the RCNS Attribution-Record extension fields in [AGTP-API]). All four fields are conditional; absence is itself diagnostic. The base schema (audit_record_version, previous_audit_id, chain semantics) is unchanged.

7. *acting_principal_id field added for external-IdP composition.* The Identifier-Chain Fields table is extended with acting_principal_id, a conditional string field carrying the identifier of the principal on whose behalf the agent acted, lifted from a validated external-IdP credential (typically the OIDC sub claim) per the Composition with External Identity Providers section in [AGTP]. The field is distinct from owner_id (which identifies the agent's permanent principal on the Agent Genesis) and from the Agent-ID: it identifies the per-request acting principal, not the agent itself or its owner. Verifiers walking the chain *SHOULD* consider acting_principal_id (when present) when reconstructing on-whose-behalf semantics. The raw credential the claim was lifted from *MUST NOT* appear on the Attribution-Record; only the validated claim does.

16.2. Wire Format Compatibility

None. The corrections are editorial: they align this document's wording with the behavior already specified in [AGTP] and deployed by v07-conformant implementations.

17. Acknowledgments

The identifier stack model was developed during the AGTP v07 revision cycle. Several reviewers and adjacent working groups shaped the architecture this document records.

Scott Courtney (GoDaddy / ANS) provided early review on the identity model and surfaced concerns about dual-logging between identity-lifecycle and per-action audit records that ultimately drove the architectural separation between [AGTP-LOG] and the per-agent Attribution-Record chain defined here.

Philip Griffiths (Cloud Security Alliance ZTCPP / ONUG AOMC) contributed feedback on zero-trust alignment that informed the verification procedure and chain-break framing.

The IETF WIMSE working group’s engagement on draft-klrc-aiagent-auth shaped the framing of agent-to-agent authentication boundaries that this document’s per-interaction identifiers sit alongside. The authors of klrc-aiagent-auth (Pieter Kasselman, Jeff Lombardo, Yaroslav Rosomakho, Brian Campbell, Nick Steele) defined the adjacent problem space.

The chain-of-custody framing draws on prior work in transparency logs ([RFC9162], [RFC9943]) and on the attestation model of [RFC9334]. The Attribution-Record envelope on which this document builds is defined in [AGTP]. The regulatory mapping was informed by the record-keeping requirements of [EU-AI-ACT].

18. References

18.1. Normative References

[AGTP] Hood, C., "Agent Transfer Protocol (AGTP)", Work in Progress, Internet-Draft, draft-hood-independent-agtp-07, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-independent-agtp-07>.

[AGTP-LOG] Hood, C., "AGTP Transparency Log Protocol", Work in Progress, Internet-Draft, draft-hood-agtp-log-02, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-log-02>.

[AGTP-SESSION] Hood, C., "AGTP Session Protocol", Work in Progress, Internet-Draft, draft-hood-agtp-session-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-session-00>.

[AGTP-TRUST]
Hood, C., "AGTP Trust and Verification Specification", Work in Progress, Internet-Draft, draft-hood-agtp-trust-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-trust-00>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/rfc/rfc7515>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC9562] Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, DOI 10.17487/RFC9562, May 2024, <https://www.rfc-editor.org/rfc/rfc9562>.

18.2. Informative References

[AGTP-API] Hood, C., "AGTP API Contract Specification", Work in Progress, Internet-Draft, draft-hood-agtp-api-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-api-00>.

[AGTP-CERT]
Hood, C., "AGTP Agent Certificate Extension", Work in Progress, Internet-Draft, draft-hood-agtp-agent-cert-01, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-agent-cert-01>.

[AGTP-MERCHANT]
Hood, C., "AGTP Merchant Identity for Agentic Commerce", Work in Progress, Internet-Draft, draft-hood-agtp-merchant-identity-02, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-merchant-identity-02>.

[EU-AI-ACT]
"Regulation (EU) 2024/1689 of the European Parliament and of the Council on artificial intelligence", 2024, <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>.

[RFC9162] Laurie, B., Messeri, E., and R. Stradling, "Certificate Transparency Version 2.0", RFC 9162, DOI 10.17487/RFC9162, December 2021, <https://www.rfc-editor.org/rfc/rfc9162>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://www.rfc-editor.org/rfc/rfc9334>.

[RFC9943] "*** BROKEN REFERENCE ***".

Author's Address

Chris Hood
Nomotic, Inc.
Email: chris@nomotic.ai
URI: https://nomotic.ai