Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 17 November 2026

T. Sato
MyAuberge K.K.

17 May 2026

The Intent Declaration Primitive (IDP) for Agentic AI Systems
draft-sato-soos-idp-01

Abstract

AI agents operating in automated workflows take actions without any normative mechanism for expressing why those actions are being taken. Access tokens declare what an agent is permitted to do; no existing standard declares what the agent believes it is doing, on what reasoning basis, and with what level of confidence, at the moment of action. This document defines the Intent Declaration Primitive (IDP): a structured per-transition declaration submitted by an AI agent to the governing OS kernel at each action step of an execution loop. The IDP is committed to a tamper-evident Event Log before the action executes, enabling post-hoc review of agent reasoning, richer authorization policy evaluation, and enriched denial responses that guide agent behaviour. The IDP also provides the technical basis for compliance with EU AI Act Article 12 logging requirements for high-risk AI systems. This document adds the IDP Commitment Verification mechanism, by which the kernel verifies at each state transition that the agent’s actual action matches its declared intent.

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 17 November 2026.

Copyright Notice

Table of Contents

1. Introduction

The Internet Engineering Task Force and related standards bodies have made significant progress in specifying how AI agents authenticate (WIMSE [I-D.ietf-wimse-arch]) and how they obtain authorization tokens for API invocation (OAuth 2.1 [I-D.ietf-oauth-v2-1], AAuth [I-D.rosenberg-oauth-aauth]). These specifications answer the question: is this agent permitted to perform this action?

No existing specification answers the companion question: why does this agent believe it should perform this action at this moment?

This distinction matters for three reasons. First, post-hoc review of AI agent behaviour requires not only a record of what actions were taken but a record of the reasoning that led to them. Access token issuance records do not provide this. Second, authorization policy evaluation can be made significantly more precise when the agent’s declared intent is available as a policy context attribute; a Cedar [Cedar] policy that evaluates both the action requested and the reasoning basis for that action is a more powerful governance tool than one that evaluates the action alone. Third, EU AI Act Article 12 requires that high-risk AI systems maintain logs sufficient for post-hoc human review of system decisions [EUAIA]; per-transition intent declaration is the technical specification that satisfies this requirement.

This document defines the Intent Declaration Primitive (IDP): a structured JSON object submitted by an AI agent to a governing runtime kernel at each action step of an agent execution loop, prior to the execution of the action. The IDP is recorded in a tamper-evident append-only Event Log before the action is executed. When authorization is denied, the kernel returns an Enriched DENY Response that includes the denial reason, available alternative actions, and the echo of the received IDP -- enabling the agent to reason about what it may do next.

The IDP is a kernel primitive, not an application-layer logging

construct. It is submitted as part of the action invocation call, not after it. This is the property that gives the IDP its audit integrity: the reasoning declaration precedes the action record, making ex-post fabrication of intent structurally impossible within a conforming implementation.

This document also specifies IDP Commitment Verification (Section 5.5): a kernel mechanism that verifies, after every governed state transition, that the agent's actual action matches its declared IDP commitment. When a mismatch is detected -- the agent declared one intent and took a different action -- the kernel generates an IDP_COMMITMENT_GAP audit finding and automatically triggers Human Escalation Mechanism Class 2 (HEM_AGENT_ESCALATED).

Changes from draft-sato-soos-idp-00:

- Section 4.1: audit_accessible field added to IDP Core Record.
- Section 5.5: IDP Commitment Verification added (new section). Defines IDP Commitment Verification Record schema, IDP_COMMITMENT_VERIFIED and IDP_COMMITMENT_GAP Event Log entries, and IDP_COMMITMENT_GAP response protocol.
- Section 11: IANA registry names corrected (removed implementation-specific prefix).

2. Conventions and Definitions

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.

This document uses the following terms:

Agent:
: A software system that uses AI, including large language models, to reason about and take actions in pursuit of goals. An agent may be human-supervised, semi-autonomous, or fully autonomous.

Governing Kernel:
: A runtime component that enforces authorization policy, records agent actions to a tamper-evident Event Log, and mediates agent access to governed objects. [I-D.sato-soos-hem] specifies the Human Escalation Mechanism that the governing kernel implements for human oversight of agent sessions.

Governed Object:
: A typed, stateful entity that the kernel governs. An agent operates on governed objects; the kernel enforces what transitions are permitted.

Mandate JWT:
: A JSON Web Token [RFC7519] that binds an agent's authorization to a specific governed object instance. The mandate JWT is the authorization credential presented alongside an IDP.

Event Log:
: An append-only, causally ordered, tamper-evident log of all transitions executed against governed objects. The Event Log is the ground truth of system behaviour.

Cedar:
: A policy language and evaluation engine [Cedar] used by the

governing kernel to evaluate authorization decisions. Cedar evaluates before any state transition executes.

HEM:
Human Escalation Mechanism. A kernel-enforced session state in which no transitions may execute until a designated human principal provides a decision. See [I-D.sato-soos-hem].

IDP:
Intent Declaration Primitive. The structured per-transition reasoning declaration defined in this document.

IDP_THIN:
A reduced IDP profile for agents with limited reasoning capability. See Section 7.

IDP Commitment Verification Record:
A kernel-generated record produced after every governed state transition, recording whether the agent's actual action matched its IDP commitment. See Section 5.5.

IDP_COMMITMENT_GAP:
A condition in which the kernel determines that the agent's actual state transition does not match the agent's declared IDP commitment. A critical audit finding. See Section 5.5.

3. Problem Statement

3.1. The Intent Gap in Agentic Authorization

Current agentic authorization frameworks model agent behaviour as a set of permitted actions, expressed as OAuth scopes, Cedar action sets, or similar constructs. This model captures what an agent may do. It does not capture, at the moment of action:

* what goal the agent believes the action serves,
* what reasoning led the agent to select this action over alternatives,
* how confident the agent is that this action is appropriate, or
* whether the agent believes human judgment should be sought before proceeding.

The absence of per-transition intent records creates four concrete problems:

(a) Audit incompleteness. Post-hoc review of AI agent behaviour requires reconstruction of agent reasoning from action records alone. This is insufficient for high-stakes environments where understanding why an action was taken is as important as knowing what action was taken.

(b) Policy imprecision. Authorization policy evaluated solely against action type and agent identity cannot distinguish between an agent acting on a direct human instruction and an agent acting on an autonomous inference. These may warrant different authorization treatment.

(c) Opaque denials. When an agent's action is denied, the agent receives no structured information about what alternative actions are available or what reasoning adjustments might lead to a permitted action. This forces agents into uninformed retry loops.

(d) Regulatory non-compliance. EU AI Act Article 12 requires logs sufficient for human review of AI system decisions for high-risk

systems. Access token issuance records do not satisfy this requirement.

3.2. Relationship to EU AI Act Article 12

EU AI Act Article 12 requires that high-risk AI systems maintain automatic logging sufficient to enable post-deployment monitoring, human review of decisions, and identification of risks. The logs must cover, at minimum, the operation of the system with reference to the relevant time period, the objects processed, and the data used.

The IDP, committed to the Event Log before each action, provides the per-decision record that Article 12 contemplates. An Event Log containing IDP entries for every agent transition produces a complete record of: what action was taken (the transition), by whom (the mandate JWT), against what (the governed object), and why (the IDP declared_goal and reasoning_basis). This document is informative with respect to regulatory compliance and does not constitute legal advice.

3.3. Relationship to Williams Intent Bound Authorization

Williams Intent Bound Authorization (IBA) [Williams-IBA] defines a human-signed authorization envelope issued before an agent session begins. The IBA envelope travels from human to agent; it constrains what the agent may do in the session.

The IDP is directionally distinct: it travels from agent to governing kernel, at each individual action step within the session. IBA expresses what the authorizing human intends the agent to do. IDP expresses what the agent declares it is doing and why.

The two primitives compose without conflict. An IBA-constrained agent session produces IDP entries at each transition; the IDP records the agent’s reasoning within the IBA-defined scope. Neither primitive replaces the other.

3.4. Relationship to Mission Bound Authorization

McGuinness Mission Bound Authorization [I-D.mcguinness-oauth-mission-bound-authorization] addresses how authorization tokens can be scoped to a declared mission, ensuring that token authority is contingent on mission validity. The IDP is directionally complementary: mission-bound authorization constrains what the agent is permitted to do within a mission; the IDP records what the agent declares it is doing and why at each individual transition within that mission. The two primitives compose without conflict. A system implementing mission-bound authorization produces IDP entries at each transition; the IDP’s mission_ref field (Section 4.1) links per-transition declarations to the governing MissionDeclaration evaluated by the mission-bound authorization framework. Neither primitive replaces the other.

4. The Intent Declaration Primitive

4.1. IDP Structure

The IDP is a JSON object with the following structure:

```
{
  "idp_id":           string,   ; REQUIRED. UUID v4.
  "session_id":       string,   ; REQUIRED. Agent session identifier.
  "so_id":            string,   ; REQUIRED. Governed object UUID.
  "mandate_id":       string,   ; REQUIRED. Mandate JWT jti claim.
```

```
  "mission_ref":      string,   ; OPTIONAL. MissionDeclaration UUID.
                                ; Links this per-transition declaration
                                ; to the governing MissionDeclaration.
                                ; MUST match the mission_ref in the
                                ; mandate JWT if that field is present.
  "step_sequence":    integer,  ; REQUIRED. ACT step number, 1-based.
  "requested_action": string,   ; REQUIRED. Cedar action string.
  "declared_goal": {            ; REQUIRED.
    "goal_id":        string,   ; REQUIRED. UUID v4.
    "description":    string    ; REQUIRED. Human-readable. Max 500 chars.
  },
  "reasoning_basis": {          ; REQUIRED.
    "type":           string,   ; REQUIRED. See Section 4.3.
    "description":    string    ; REQUIRED. Human-readable. Max 1000 chars.
  },
  "confidence_level": number,   ; REQUIRED. Float, 0.0 to 1.0 inclusive.
  "hem_urgency":      string,   ; REQUIRED. See Section 4.4.
  "context_refs":    [string],  ; OPTIONAL. UUIDs of relevant prior events.
  "audit_accessible": boolean,  ; OPTIONAL. Default: true. When false,
                                ; IDP is not accessible to Audit
                                ; Principals via kernel query interfaces.
  "metadata":         object,   ; OPTIONAL. Implementation-defined.
  "timestamp":        string    ; REQUIRED. ISO 8601 UTC.
}
```

4.2. Field Definitions

idp_id:
: A UUID v4 [RFC4122] assigned by the agent. Uniquely identifies this IDP within the Event Log. MUST be unique across all IDPs for the lifetime of a governed object instance.

session_id:
: The identifier of the agent session within which this action is taken. MUST match the session_id established at session creation.

so_id:
: The UUID v4 of the governed object being operated on. MUST match the UUID bound to the mandate_id.

mandate_id:
: The jti claim of the mandate JWT presented with this IDP. Links the intent declaration to the authorization credential under which the action is requested.

mission_ref:
: OPTIONAL. The UUID of the MissionDeclaration governing the session in which this IDP is submitted. When present, this field links the per-transition intent declaration to the overarching mission under which the agent is operating. If the mandate JWT carries a mission_ref claim, the IDP mission_ref MUST match that claim. Absent if the session does not operate under a declared MissionDeclaration.

step_sequence:
: A positive integer indicating the ordinal position of this ACT step within the current session. Starts at 1. MUST be monotonically increasing within a session. The kernel SHOULD reject an IDP with a step_sequence equal to or less than the last committed step_sequence for this session.

requested_action:
: The Cedar action string identifying the action the agent requests. MUST be the same Cedar action string submitted to the kernel for policy evaluation.

declared_goal:
A structured declaration of the goal the agent believes this action serves.

goal_id:
A UUID v4 assigned by the agent. MAY be shared across multiple IDPs within a session if those IDPs are understood to serve the same goal.

description:
A human-readable description of the goal. MUST NOT contain personally identifiable information. Maximum 500 characters.

reasoning_basis:
A structured declaration of the reasoning that led the agent to request this action.

type:
One of the reasoning basis types defined in Section 4.3.

description:
A human-readable description of the specific reasoning. MUST NOT contain personally identifiable information. Maximum 1000 characters.

confidence_level:
A floating-point number in [0.0, 1.0] expressing the agent's self-assessed confidence that this action is appropriate. 1.0 indicates full confidence; 0.0 indicates no confidence. This value is self-reported by the agent and is not independently verified by the kernel. Implementers SHOULD treat this value as a declaration, not a guarantee.

hem_urgency:
A declaration of the agent's assessment of whether human judgment should be sought. See Section 4.4.

context_refs:
An OPTIONAL array of Event Log entry UUIDs that the agent considers relevant to its reasoning. Allows tracing of how prior events informed the current action.

audit_accessible:
OPTIONAL. Boolean. Default: true. When true, this IDP record is accessible to Audit Principals via kernel query interfaces (kernel.query_event_log()) and is included in Session Audit Records and Audit Packages. When false, the IDP is accessible only to the governing kernel's internal audit functions and to Verified External Auditors; it is not returned in response to Audit Principal queries. Operators who require restricted IDP access for confidentiality reasons MAY set this field to false; such restrictions MUST be documented in the Session Audit Record.

metadata:
An OPTIONAL implementation-defined JSON object. The kernel MUST record this field in the Event Log without interpretation. Domain-specific IDP extensions SHOULD use this field.

timestamp:
The time at which the agent generated the IDP, in ISO 8601 format with UTC timezone. The kernel MUST record both this timestamp and the kernel receipt timestamp in the Event Log entry.

4.3. Reasoning Basis Types

The reasoning_basis.type field MUST be one of the following values:

RULE_BASED:
The agent is executing a predetermined rule or procedure. The action was selected by applying a defined decision rule to the current state, not by open-ended inference. Example: "Booking confirmation is required when payment is received; payment has been received."

INFERENCE:
The agent inferred that this action is appropriate based on contextual reasoning. The action was not explicitly prescribed by a rule; the agent applied judgment to conclude it is correct. Example: "Based on the stated guest preferences and current availability, this room assignment appears optimal."

INSTRUCTION:
The agent is executing an explicit instruction from a human principal or from a higher-authority agent in the delegation chain. Where the mandate JWT carries a delegation_chain claim [I-D.mcguinness-oauth-actor-profile], the source of the instruction is traceable through the full actor chain. The reasoning_basis.description MUST identify the source of the instruction by mandate_id or session_id.

UNCERTAINTY_REDUCTION:
The agent is taking this action to reduce uncertainty, not to directly advance the declared goal. Used when an agent performs a read or query action to gather information before determining the appropriate forward action.

MISSION_STAGE:
The agent is executing an action whose primary purpose is to advance the governing MissionDeclaration from one stage to the next. This type is used when the mission stage transition, not the domain goal itself, is the proximate reason for the action. The mission_ref field MUST be present when this type is used.

Implementations MAY define additional type values. Such values MUST be registered in the IDP Reasoning Basis Types registry (see Section 11) or prefixed with a URI identifying the defining organization. Unrecognized type values MUST be recorded in the Event Log and MUST NOT cause the kernel to reject the IDP.

4.4. HEM Urgency Values

The hem_urgency field is a declaration by the agent of its assessment of whether human judgment should be sought before this action executes. The kernel uses this value as one input to HEM trigger determination.

NONE:
The agent assesses that no human judgment is needed for this action. The kernel proceeds with normal Cedar evaluation.

RECOMMENDED:
The agent assesses that human judgment would be beneficial but does not consider it required. The kernel SHOULD record this value but is not required to initiate HEM. Kernel implementations MAY use this value to trigger HEM based on local policy.

REQUIRED:
The agent declares that it requires human judgment before this action executes. The kernel MUST initiate HEM_PENDING for this session upon receipt of an IDP with hem_urgency: REQUIRED, regardless of Cedar policy evaluation outcome. This corresponds

to the HEM_AGENT_ESCALATED trigger class defined in [I-D.sato-soos-hem].

Note: A hem_urgency value of REQUIRED from the agent does not override kernel Cedar evaluation. Cedar evaluation still executes. However, even a Cedar PERMIT result MUST NOT cause the action to execute; the session enters HEM_PENDING, and the action awaits human decision.

4.5. IDP Profiles

This document defines two IDP profiles:

IDP_STANDARD:
All REQUIRED fields present. confidence_level is agent-assessed. reasoning_basis.type is one of the defined values. This is the normative profile for agents with sufficient reasoning capability to self-assess.

IDP_THIN:
A reduced profile for agents with limited reasoning capability. See Section 7.

The kernel MUST accept both profiles. The kernel MUST record the profile type in the IDP Event Log entry.

5. Submission Protocol

5.1. When the IDP Is Submitted

The IDP MUST be submitted by the agent as part of the same kernel call that requests the state transition. The IDP MUST NOT be submitted after the action has executed.

In a conforming agentic execution loop, the sequence is:

1. (SENSE) Kernel delivers context to agent.
2. (REASON) Agent reasons about next action.
3. (PLAN) Agent consults Transition Graph API for available actions.
4. (ACT) Agent submits kernel.transition(mandate_jwt, cedar_action, idp) -- the IDP is part of this call.
5. (OBSERVE) Agent receives result: success, enriched DENY, or SESSION_HEM_PENDING.
6. (LOOP) Agent processes result before next ACT.

The IDP MUST be present in the kernel.transition() call. A kernel.transition() call without an IDP MUST be rejected by a conforming kernel implementation.

5.2. Kernel Receipt and Validation

Upon receiving a kernel.transition() call, the kernel MUST:

(a) Validate that an IDP is present. If absent, return REJECT with error code IDP_MISSING.

(b) Validate that the IDP fields conform to this specification. If malformed, return REJECT with error code IDP_MALFORMED.

(c) Validate that the idp_id is unique for this governed object. If a duplicate idp_id is detected, return REJECT with error code IDP_DUPLICATE.

(d) Validate that the IDP so_id matches the governed object UUID bound to the mandate_jwt. If mismatched, return REJECT with

error code IDP_SO_MISMATCH.

(e) Validate that the IDP mandate_id matches the jti claim of the presented mandate_jwt. If mismatched, return REJECT with error code IDP_MANDATE_MISMATCH.

(f) Validate that step_sequence is strictly greater than the last committed step_sequence for this session.

(g) If mission_ref is present, validate that it matches the mission_ref claim in the mandate_jwt (if that claim is present). If mismatched, return Enriched DENY with deny_code IDP_MISSION_REF_MISMATCH and a structured mismatch_detail block containing the expected and submitted mission_ref values. The kernel MUST log IDP_MISSION_REF_MISMATCH_REJECTED in the Event Log. If mission_ref mismatches exceed a configurable threshold within a session, the kernel SHOULD automatically trigger HEM_AGENT_ESCALATED (Class 2).

Kernel validation of the IDP precedes Cedar evaluation. A valid IDP is required for Cedar evaluation to proceed.

5.3. Event Log Commitment

A conforming kernel implementation MUST commit an IDP_SUBMITTED event to the Event Log upon successful validation of the IDP, BEFORE Cedar evaluation executes.

The IDP_SUBMITTED event MUST contain:

* The full IDP object as submitted.
* The kernel receipt timestamp (distinct from the agent timestamp).
* The mandate_id.
* The session_id.
* The audit_accessible value (defaults to true if absent).
* The kernel signature over the event.

The IDP_SUBMITTED event is committed regardless of the subsequent Cedar evaluation outcome. If Cedar evaluation results in DENY, the IDP_SUBMITTED event remains in the Event Log as a permanent record of the agent’s declared intent.

The Event Log ordering guarantee is:

```
IDP_SUBMITTED < Cedar evaluation < STATE_TRANSITIONED
(or CEDAR_DENY_RECORDED)
```

No STATE_TRANSITIONED event may appear in the Event Log without a preceding IDP_SUBMITTED event for the same step_sequence.

Following every STATE_TRANSITIONED event, the kernel MUST generate an IDP_COMMITMENT_VERIFIED or IDP_COMMITMENT_GAP event as specified in Section 5.5.

5.4. IDP in Cedar Policy Evaluation

The kernel MUST make the following IDP fields available as Cedar context attributes during policy evaluation:

```
* idp.reasoning_basis.type  -- string
* idp.confidence_level      -- decimal
* idp.hem_urgency           -- string
* idp.goal_id               -- string
* idp.mission_ref           -- string (null if absent)
```

Cedar policies MAY use these attributes in condition expressions.

Example policies:

```
// Require INSTRUCTION basis for high-value transitions
permit(principal, action == Action::"ProcessPayment", resource)
when {
  context.idp.reasoning_basis.type == "INSTRUCTION"
};

// Require confidence >= 0.8 for autonomous state closure
permit(principal, action == Action::"CloseBooking", resource)
when {
  context.idp.confidence_level >= decimal("0.8")
};

// Require mission_ref for MISSION_STAGE transitions
permit(principal, action == Action::"AdvanceMissionStage", resource)
when {
  context.idp.reasoning_basis.type == "MISSION_STAGE" &&
  context.idp.mission_ref != null
};
```

Cedar policies that reference IDP attributes but receive an IDP_THIN submission (Section 7) where the relevant attribute is absent MUST treat the absent attribute as if it had a value that causes the condition to evaluate to false, resulting in DENY. This ensures that thin-profile agents cannot satisfy policies written for standard-profile agents by omission.

5.5. IDP Commitment Verification

After every successful governed state transition (STATE_TRANSITIONED Event Log entry), the kernel MUST generate an IDP Commitment Verification Record and commit it to the Event Log. The purpose of Commitment Verification is to detect IDP_COMMITMENT_GAP conditions: cases where the agent's actual state transition does not match the action declared in its preceding IDP.

This mechanism enforces the non-suppressibility property: the kernel cannot conceal the fact that an agent's actual behaviour diverged from its declared intent.

5.5.1. IDP Commitment Verification Record Schema

The IDP Commitment Verification Record MUST contain the following fields:

idp_id:
: The idp_id of the IDP submitted for this transition. Links the verification record to the specific intent declaration.

state_transition_id:
: The Event Log identifier of the STATE_TRANSITIONED entry being verified against.

verified_at:
: ISO 8601 UTC timestamp of verification. Generated by the kernel.

match_result:
: One of:

  MATCHED: The Cedar action string in the IDP's requested_action field matches the action that was executed in the state transition. The agent acted in accordance with its declaration.

  IDP_COMMITMENT_GAP: The Cedar action string in the IDP's requested_action field does not match the action that was

executed. This is a critical audit finding.

kernel_signature:
Ed25519 signature over the canonical serialization of all fields except kernel_signature.

5.5.2. Event Log Entries

IDP_COMMITMENT_VERIFIED:
Committed when match_result is MATCHED. Contains the full IDP Commitment Verification Record.

IDP_COMMITMENT_GAP:
Committed when match_result is IDP_COMMITMENT_GAP. Contains the full IDP Commitment Verification Record. This entry MUST be treated as a critical audit finding. The kernel MUST immediately:

(a) Generate a CRITICAL Audit Alert (alert_trigger: IDP_COMMITMENT_GAP).

(b) Fire HEM_AGENT_ESCALATED (Class 2 per [I-D.sato-soos-hem] Section 5.2) for the active session. The HEM Escalation Request MUST include the IDP Commitment Verification Record in the idp_summary, with match_result: IDP_COMMITMENT_GAP clearly surfaced to the resolving principal.

The kernel MUST NOT allow the agent session to continue after an IDP_COMMITMENT_GAP without HEM resolution. The session enters HEM_PENDING immediately.

5.5.3. Commitment Verification Scope

The kernel compares the requested_action field of the IDP against the Cedar action string of the executed transition. The comparison is string-exact. A mismatch on any character MUST be recorded as IDP_COMMITMENT_GAP.

IDP_COMMITMENT_GAP does not distinguish between:

- An agent that declared one action and executed a different one due to an implementation error.
- An agent that was coerced or compromised after IDP submission.
- An agent that exploited a race condition between IDP submission and transition execution.

All three produce the same audit finding. The distinction is a matter for human principal investigation following HEM resolution.

5.5.4. Relationship to GAR

The IDP Commitment Verification Record and its Event Log entries are specified in this document. Their inclusion in Session Audit Records and Audit Packages is governed by [I-D.sato-soos-gar]. The IDP_COMMITMENT_GAP Audit Alert trigger is registered in the GAR Audit Alert Triggers registry.

6. Enriched DENY Response

6.1. Structure

When Cedar evaluation results in DENY, a conforming kernel implementation MUST return an Enriched DENY Response rather than a bare authorization failure. The Enriched DENY Response is a JSON

object:

```
{
  "result":            "DENY",   ; REQUIRED.
  "deny_code":         string,   ; REQUIRED. See Section 6.2.
  "deny_reason":       string,   ; REQUIRED. Human-readable.
  "idp_received":      object,   ; REQUIRED. Echo of submitted IDP.
  "available_actions": [string], ; REQUIRED. Cedar actions currently
                                 ; permitted for this agent and SO.
  "suggested_paths":   [object], ; OPTIONAL. See Section 6.2.
  "hem_available":     boolean,  ; REQUIRED.
  "mismatch_detail":   object,   ; CONDITIONAL. Present when deny_code
                                 ; is IDP_MISSION_REF_MISMATCH.
  "timestamp":         string    ; REQUIRED. ISO 8601 UTC.
}
```

6.2. Field Definitions

deny_code:
A machine-readable denial code. The following codes are defined:

POLICY_DENY	Cedar policy denied the action.
MANDATE_SCOPE	Action is outside the mandate's Cedar scope.
SO_STATE_INVALID	Action is not valid in the current SO state.
HEM_PENDING	Session is in HEM_PENDING; no transitions permitted until HEM is resolved.
MANDATE_REVOKED	Mandate JWT has been revoked.
CROSS_PRINCIPAL_UNRESOLVED	Cross-principal coordination required.
MISSION_INVALID	The governing MissionDeclaration is in a terminal phase (SUSPENDED, FAILED, or ABANDONED). No transitions are permitted until the mission state is resolved.
IDP_MISSION_REF_MISMATCH	IDP mission_ref does not match the active session MissionDeclaration. See mismatch_detail.

deny_reason:
A human-readable explanation of why the action was denied. SHOULD be specific enough to guide the agent's next reasoning step. MUST NOT expose internal policy structure that would enable policy circumvention.

idp_received:
The complete IDP object as received and validated by the kernel. Allows the agent to verify that its declaration was received accurately.

available_actions:
An array of Cedar action strings that the agent's current mandate permits in the current SO state. MAY be empty if no actions are currently available. This is the primary mechanism by which the kernel guides agents away from prohibited actions toward permitted ones.

suggested_paths:
An OPTIONAL array of path objects, each representing a sequence of actions that would lead the agent to a state where the denied action becomes available. Each path object contains:

```
{
  "path_id":          string,  ; UUID.
  "steps":           [string], ; Ordered Cedar action strings.
  "requires_elevation": boolean
```

```
}
```

hem_available:
A boolean indicating whether the agent may trigger HEM by resubmitting with hem_urgency: REQUIRED. False if the session is already in HEM_PENDING or if HEM is not configured for this SO Type.

mismatch_detail:
CONDITIONAL. Present when deny_code is IDP_MISSION_REF_MISMATCH. Contains:

```
{
  "expected_mission_ref": string,  ; mission_ref from mandate JWT.
  "submitted_mission_ref": string  ; mission_ref from IDP.
}
```

Enables the submitting agent to diagnose and correct the mismatch without ambiguity.

6.3. HEM Routing from DENY

A Cedar DENY result with deny_code POLICY_DENY does not automatically trigger HEM. However, the agent MAY respond to a DENY by resubmitting the same action with hem_urgency: REQUIRED. If hem_available is true in the Enriched DENY Response, the kernel MUST accept the resubmission and enter HEM_PENDING, routing the decision to a designated human principal.

This provides a normative escalation path for situations where the agent’s Cedar-authorized scope does not include the needed action but the agent has grounds to believe that human judgment may permit it.

7. IDP Thin Profile

Some agents -- particularly narrow executors operating on simple tasks -- may have limited capability to produce full IDP_STANDARD declarations. The IDP_THIN profile accommodates these agents while maintaining the core audit property of the IDP.

An IDP_THIN submission MUST contain:

* idp_id (REQUIRED)
* session_id (REQUIRED)
* so_id (REQUIRED)
* mandate_id (REQUIRED)
* step_sequence (REQUIRED)
* requested_action (REQUIRED)
* profile: "IDP_THIN" (REQUIRED; absent in IDP_STANDARD)
* timestamp (REQUIRED)

An IDP_THIN submission MUST omit:

* declared_goal (OPTIONAL in IDP_THIN; kernel synthesizes stub)
* reasoning_basis (OPTIONAL in IDP_THIN; kernel records "UNSPECIFIED")
* confidence_level (OPTIONAL in IDP_THIN; kernel records 0.5 default)
* hem_urgency (OPTIONAL in IDP_THIN; kernel defaults to "NONE")
* mission_ref (OPTIONAL in IDP_THIN; kernel records null)

The audit_accessible field is OPTIONAL in IDP_THIN; kernel defaults to true if absent.

The kernel MUST record the IDP_THIN profile in the IDP_SUBMITTED event. The kernel SHOULD synthesize stub values for absent fields

to maintain Event Log schema consistency.

A conforming SO Type MAY declare that IDP_THIN submissions are not accepted for transitions of a given type. In this case, a kernel receiving an IDP_THIN submission for a prohibited transition MUST return REJECT with error code IDP_THIN_NOT_ACCEPTED.

IDP_THIN submissions are subject to IDP Commitment Verification (Section 5.5). The requested_action field is present in IDP_THIN and is the basis for the commitment comparison.

Note: This specification identifies thin profile kernel compensation as an area for future work. Mechanisms by which the kernel may synthesize richer IDP content from contextual signals for thin-profile agents are outside the scope of this document.

8. Security Considerations

8.1. IDP as Declaration, Not Proof

The IDP records the agent's declared intent; it does not verify it. A compromised or adversarial agent may submit an IDP that does not accurately represent its actual reasoning. The security value of the IDP lies not in proof of intent but in the permanent, pre-action record that creates accountability: an agent that acts contrary to its declared intent produces an auditable inconsistency between the IDP record and the observed action sequence. The IDP Commitment Verification mechanism (Section 5.5) detects this inconsistency automatically.

8.2. Tamper Evidence

The IDP_SUBMITTED Event Log entry MUST be kernel-signed before Cedar evaluation executes. This prevents retroactive modification of the intent record. Implementations MUST use an asymmetric signing key held by the kernel attestation component and MUST NOT permit agents to write directly to the Event Log.

8.3. Inference from Denials

The Enriched DENY Response reveals information about permitted actions and authorization policy structure. Implementations MUST ensure that available_actions and suggested_paths do not reveal policy conditions that would enable adversarial agents to reverse-engineer policy bypass strategies. The deny_reason field MUST be informative without being exploitable.

8.4. Replay Prevention

The idp_id uniqueness requirement (Section 5.2(c)) prevents replay of previously committed IDP objects. Implementations MUST maintain an in-memory index of committed idp_ids for the lifetime of each governed object, rebuilt from the Event Log on kernel restart.

8.5. Delegation Chain Integrity

The IDP's mandate_id field links the intent declaration to the authorization credential. In multi-hop delegation chains, the mandate_id identifies the specific delegation under which the action is requested. This enables audit tracing of intent across delegation trees.

In systems implementing the actor chain model [I-D.mcguinness-oauth-actor-profile], the mandate JWT carries a delegation_chain claim identifying each principal in the issuance

chain. The IDP's mandate_id field, which references the jti of the presented mandate JWT, therefore provides a complete traceability link from the per-transition intent declaration to the full delegation chain. Audit tools reconstructing agent reasoning SHOULD traverse the delegation_chain of the referenced mandate JWT to identify all principals in the chain of authority.

8.6. Thin Profile Risk

IDP_THIN submissions provide reduced accountability because the reasoning record is minimal. High-value or high-risk transitions SHOULD require IDP_STANDARD submissions. SO Type designers SHOULD use the IDP_THIN_NOT_ACCEPTED mechanism (Section 7) for transitions where reasoning accountability is critical.

8.7. Commitment Verification Integrity

The IDP Commitment Verification Record (Section 5.5) is kernel-generated and kernel-signed. Agents MUST NOT be able to influence the match_result field. Implementations MUST ensure that the comparison between requested_action and the executed transition action string occurs entirely within the kernel, with no agent-accessible interface to the comparison logic or its result.

9. Privacy Considerations

The IDP contains agent-generated text in the declared_goal. description and reasoning_basis.description fields. These fields MUST NOT contain personally identifiable information. The kernel SHOULD validate that these fields do not contain obvious PII markers (email addresses, phone numbers, names), but CANNOT guarantee the absence of PII through syntactic validation alone.

Implementers MUST ensure that Event Log access controls prevent unauthorized parties from reading IDP records. The audit_accessible field (Section 4.1) provides a mechanism for restricting Audit Principal access to individual IDP records. IDP records are governance data, not publicly accessible telemetry.

IDP records associated with governed objects that are subject to data protection erasure requests present a tension with the Event Log's append-only property. This tension is addressed through the Zone A personal data prohibition and cryptographic erasure provisions specified in the governing kernel implementation.

10. EU AI Act Applicability

This section is informative.

EU AI Act Article 12 requires that high-risk AI systems maintain automatic logging sufficient for post-deployment monitoring and human review of AI decisions. The IDP, committed to a kernel-signed Event Log before each agent action, provides the per-decision record that Article 12 contemplates.

EU AI Act Article 14 requires human oversight measures enabling intervention in the operation of high-risk AI systems. The HEM escalation path triggered by hem_urgency: REQUIRED, and the HEM routing from Cedar DENY responses (Section 6.3), provide the normative intervention mechanism that Article 14 requires. The HEM protocol is fully specified in [I-D.sato-soos-hem].

The IDP Commitment Verification mechanism (Section 5.5) contributes to Article 14(4)(c) compliance: automatic detection of anomalous

situations (commitment gaps between declared intent and actual action) with mandatory human escalation.

This document does not constitute legal advice and makes no representation that conforming implementations satisfy any regulatory requirement.

11. IANA Considerations

This document requests the creation of the following IANA registry:

Registry Name:
IDP Reasoning Basis Types

Registration Procedure:
Specification Required [RFC8126]

Initial Values:

Type Value	Description
RULE_BASED	Action selected by applying a defined rule to the current state
INFERENCE	Action inferred from contextual reasoning
INSTRUCTION	Action executing an explicit instruction from a human principal or higher-authority agent
UNCERTAINTY_REDUCTION	Action taken to reduce uncertainty before selecting a goal-advancing action
MISSION_STAGE	Action advancing the governing MissionDeclaration to the next stage; mission_ref MUST be present

Table 1: Initial IDP Reasoning Basis Types Registry Values

This document also requests the creation of the following IANA registry:

Registry Name:
IDP Deny Codes

Registration Procedure:
Specification Required [RFC8126]

Initial Values:

Deny Code	Description
POLICY_DENY	Cedar policy denied the action
MANDATE_SCOPE	Outside mandate Cedar scope
SO_STATE_INVALID	Invalid in current SO state
HEM_PENDING	Session in HEM_PENDING state
MANDATE_REVOKED	Mandate JWT revoked
CROSS_PRINCIPAL_UNRESOLVED	Cross-principal pending
MISSION_INVALID	MissionDeclaration in terminal phase; no transitions permitted
IDP_MISSION_REF_MISMATCH	IDP mission_ref mismatch; see mismatch_detail
IDP_MISSING	IDP absent from call
IDP_MALFORMED	IDP failed schema validation
IDP_DUPLICATE	Duplicate idp_id detected
IDP_SO_MISMATCH	IDP so_id / mandate mismatch

| IDP_MANDATE_MISMATCH | IDP mandate_id / JWT mismatch |
| IDP_THIN_NOT_ACCEPTED | IDP_THIN not accepted here |

Table 2: Initial IDP Deny Codes Registry Values

12. References

12.1. Normative References

[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>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <https://www.rfc-editor.org/rfc/rfc4122>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/rfc/rfc7519>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/rfc/rfc8126>.

[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>.

[Cedar] Amazon Web Services, "Cedar Policy Language", <https://www.cedarpolicy.com/>.

[I-D.sato-soos-hem] Sato, T., "The Human Escalation Mechanism (HEM) for Agentic AI Systems", Work in Progress, Internet-Draft, draft-sato-soos-hem-00, May 2026, <https://datatracker.ietf.org/doc/draft-sato-soos-hem/>.

12.2. Informative References

[EUAIA] European Union, "Regulation (EU) 2024/1689 of the European Parliament and of the Council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)", Official Journal of the European Union, 12 July 2024, <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L_202401689>.

[I-D.ietf-wimse-arch] Salowey, J., Rosomakho, Y., and H. Tschofenig, "Workload Identity in a Multi System Environment (WIMSE) Architecture", Work in Progress, Internet-Draft, draft-ietf-wimse-arch-07, March 2026, <https://datatracker.ietf.org/doc/draft-ietf-wimse-arch/>.

[I-D.ietf-oauth-v2-1] Hardt, D., Parecki, A., and T. Lodderstedt, "The OAuth 2.1 Authorization Framework", Work in Progress, Internet-Draft, draft-ietf-oauth-v2-1, 2026, <https://datatracker.ietf.org/doc/draft-ietf-oauth-v2-1/>.

[I-D.klrc-aiagent-auth]
Kasselman, P., Lombardo, J., Rosomakho, Y., Campbell, B., and N. Steele, "AI Agent Authentication and Authorization", Work in Progress, Internet-Draft, draft-klrc-aiagent-auth-01, March 2026, <https://datatracker.ietf.org/doc/draft-klrc-aiagent-auth/>.

[I-D.rosenberg-aiproto-cheq]
Rosenberg, J., White, P., and C. Jennings, "CHEQ: A Protocol for Confirmation AI Agent Decisions with Human in the Loop (HITL)", Work in Progress, Internet-Draft, draft-rosenberg-aiproto-cheq-00, October 2025, <https://datatracker.ietf.org/doc/draft-rosenberg-aiproto-cheq/>.

[I-D.sato-soos-gar]
Sato, T., "The Governance Audit Record (GAR) for Agentic AI Systems", Work in Progress, Internet-Draft, draft-sato-soos-gar-00, May 2026, <https://datatracker.ietf.org/doc/draft-sato-soos-gar/>.

[I-D.sato-soos-transition-graph]
Sato, T., "Transition Graph API for Agentic AI Systems", Work in Progress, Internet-Draft, draft-sato-soos-transition-graph-00, 2026 (forthcoming).

[I-D.mcguinness-oauth-actor-profile]
McGuinness, K., "OAuth Actor Profile for AI Agents", Work in Progress, Internet-Draft, draft-mcguinness-oauth-actor-profile-00, 2026, <https://datatracker.ietf.org/doc/draft-mcguinness-oauth-actor-profile/>.

[I-D.mcguinness-oauth-mission-bound-authorization]
McGuinness, K., "Mission Bound Authorization", Work in Progress, Internet-Draft, draft-mcguinness-oauth-mission-bound-authorization-00, 2026, <https://datatracker.ietf.org/doc/draft-mcguinness-oauth-mission-bound-authorization/>.

[Williams-IBA]
Williams, J., "Intent Bound Authorization", GB2603013.0, PCT pending, 2025.

Acknowledgments

The IDP design draws on the Windley Loop context injection model and the Cedar policy evaluation architecture [Cedar]. The directional distinction from Williams IBA was identified through review of [Williams-IBA]. The mission_ref field and MISSION_STAGE reasoning type were added following review of the McGuinness Mission Bound Authorization framework [I-D.mcguinness-oauth-mission-bound-authorization] and the actor chain model [I-D.mcguinness-oauth-actor-profile]. The IDP Commitment Verification mechanism was developed in conjunction with the Governance Audit Record specification [I-D.sato-soos-gar].

Author’s Address

Tom Sato
MyAuberge K.K.
Chino, Nagano

Japan
Email: tomsato@myauberge.jp
URI: https://activitytravel.pro/