Individual Submission
Internet-Draft
Intended status: Informational
Expires: 15 November 2026

T. Pidlisnyi
AEOESS

14 May 2026

Agent Passport System (APS): Cryptographic Identity, Faceted Authority Attenuation, and Governance for AI Agent Systems
draft-pidlisnyi-aps-01

Abstract

This document specifies the Agent Passport System (APS), a protocol for cryptographic identity, faceted authority attenuation, and governance for AI agent systems. APS introduces Ed25519-based agent passports, scoped delegation chains with monotonic narrowing across seven constraint dimensions (scope, spend, depth, time, reputation, values floor, reversibility), cascade revocation, a three-signature policy chain (intent, evaluation, receipt), and signed receipts that record what an agent declared, what a policy engine decided, and what an enforcement boundary observed. Authority is modeled as an element of a product lattice, and delegation is a monotone function on that lattice, ensuring that delegated capabilities can only be attenuated, never amplified. APS also defines key rotation with identity continuity, and recognizes institutional governance primitives (charters, offices, approval policies, federation) that compose with the delegation lattice; their normative specification is companion work. The protocol is intended to complement current AI agent infrastructure, including MCP and A2A, with cryptographic identity, delegated authority, and verifiable enforcement evidence. A protocol binding is specified for MCP. Reference implementations are available in TypeScript and Python under Apache-2.0; see the Implementation Status section.

Status of This Memo

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

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

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

This Internet-Draft will expire on 15 November 2026.

Table of Contents

1. Introduction

AI agent systems are increasingly deployed in architectures where orchestrators decompose tasks and delegate subtasks to specialist agents. Existing agent communication protocols, including the Model Context Protocol (MCP) and the Agent-to-Agent Protocol (A2A), address connectivity and tool invocation. Deployments often need additional layers for cryptographic agent identity, delegated authority, and verifiable enforcement evidence. When an orchestrator delegates to a specialist that calls a tool, the delegation chain that led to the tool invocation is lost.

APS fills this gap by providing: (1) Ed25519 cryptographic identity bound to cryptographically verifiable passports; (2) scoped delegation chains where authority narrows monotonically across seven constraint dimensions; (3) cascade revocation where revoking any delegation invalidates all descendants; (4) a three-signature policy chain binding intent to evaluation to receipt; (5) signed receipts that make each stage of a governed action verifiable after the fact; (6) institutional governance primitives for multi-agent organizations; and (7) an enforcement gateway that serves as an external reference monitor.

Agentic work raises several distinct attribution questions: under whose authority an action was taken, what sources contributed to a deliverable, on whose behalf the agent acted, and who receives the value the work creates. This document specifies the authority core and the signed receipt layer through which principal resolution is recorded and other attribution axes may be referenced; it does not define contribution-attribution or beneficiary-attribution models.

The protocol's formal invariants and design rationale are published in the informative references.

1.1. 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. Identity Scheme

2.1. Agent Passport

Each agent in APS possesses an Agent Passport: a signed document binding an Ed25519 public key to an agent identifier, name, owner, and time-to-live. The passport is self-signed by the agent's private key, establishing cryptographic identity without a central authority.

2.2. DID Scheme

APS defines a DID method "did:aps" using multibase-encoded Ed25519 public keys: did:aps:z<base58btc-encoded-public-key>.

2.3. Key Rotation and Identity Continuity

An agent MAY rotate its signing key. Rotation produces a new Ed25519 key pair while preserving the agent's identity: the agent identifier and the passport's binding to an owner persist across the rotation. The agent's DID document records the rotation, so that a verifier resolving the identity can observe the key history.

A rotated key carries a retirement time. A retired key remains valid for verifying signatures produced before its retirement time, and MUST NOT be accepted for signatures produced after it. This preserves the verifiability of historical receipts and delegations signed under a key that has since been rotated out.

The protocol distinguishes a planned rotation, performed on a schedule or policy, from an emergency rotation, performed in response to suspected key compromise. The two differ in operational urgency and in how aggressively a verifier should treat signatures near the retirement boundary; they do not differ in the cryptographic mechanism.

Determining whether a signature was produced before or after a key's retirement time requires a trusted notion of when the signature was made. This document does not define a trusted timestamping service. A verifier evaluates retired-key validity against the signed timestamps carried by the receipt or delegation (for example, issued_at) together with any profile-defined trusted-timestamping or log-inclusion mechanism; absent such a mechanism, a verifier MUST treat signatures near the retirement boundary as it would any other claim that depends on issuer-asserted time.

3. Delegation and Authority Attenuation

3.1. Faceted Authority Attenuation

Agent authority is modeled as an element of a product lattice A = D_1 x D_2 x ... x D_7, where each D_k is a bounded partially ordered set. The seven dimensions are: Scope (power set, subset ordering), Spend (non-negative reals), Depth (naturals), Time (TTL seconds), Reputation (the closed interval from 0 to 100), Values (a values floor, defined below), Reversibility ({Tentative, Compensable, Irreversible}).

The Values dimension is modeled as a values floor: a finite set of attested principle identifiers required of the agent by the delegation. Its ordering is by set inclusion, with narrowing in the superset direction. A delegation MUST preserve every principle identifier required by any ancestor and MAY add further required identifiers; it MUST NOT remove an ancestor-required identifier. Formally, if V_parent is the parent's required-principle set and V_child is the child's, a valid delegation requires V_parent is a subset of V_child. A larger values floor is a narrower authority position: more required principles constrain the agent further. The Values dimension orders sets of principle identifiers; it does not order principles against one another, and this document does not define the meaning of any individual principle identifier.

Delegation is a monotone function on this lattice: for any delegation d with parent p, auth(d) <= auth(p) in the product ordering. Authority narrows monotonically along any delegation chain across all seven dimensions simultaneously. The dimensions do not all narrow in the same set-theoretic direction: Scope narrows under subset ordering while the Values floor narrows under superset ordering. Both are valid component orderings, and the monotone-narrowing property holds across the product because the product ordering is componentwise. The lattice formalization and its monotonic-narrowing property are developed in [APS-NARROWING] and [APS-FACETED].

3.2. Cascade Revocation

Any delegation MAY be revoked by its issuer. Revocation MUST cascade to all transitive descendants. Revocation is irreversible. The enforcement gateway MUST recheck revocation status at execution time, not only at approval time.

3.3. Core Invariants

The protocol specifies eight invariants: INV-1 (Identity Verifiability), INV-2 (Scope Monotonic Narrowing), INV-3 (Spend Limit Narrowing), INV-4 (Cascade Completeness), INV-5 (Revocation Irreversibility), INV-6 (Intent-Receipt Binding), INV-7 (Authority Attribution Completeness), INV-8 (Signature Integrity). INV-6 is elaborated by the signed receipt layer specified in Section 5. INV-7 (Authority Attribution Completeness) is the requirement that every governed action is attributable to an acting agent, an authority path, and a receipt context; it does not concern contribution-attribution or beneficiary-attribution, which are out of scope for this document (see Attribution Axes and Scope).

4. Policy Chain

APS defines a three-signature policy chain: ActionIntent (agent declares intended action), PolicyDecision (policy engine evaluates with verdict allow/deny/escalate), PolicyReceipt (enforcement gateway records execution result). The policy engine splits into a deterministic gate (scope, signature, revocation, authority path, spend) and an advisory evaluation path (deception, proportionality). The structure and verification of the signed records produced at each stage are specified in Section 5.

4.1. Action Reference Computation

Each action in the policy chain is identified by a deterministic action reference (action_ref) that serves as a cross-engine correlation anchor. The action_ref is computed as:

action_ref = SHA-256(canonicalize(input_object))

where input_object is a JSON object with exactly four fields:

agentId The DID of the acting agent (e.g., "did:aps:z6Mk..."), in the canonical string form defined by its DID method. For did:aps (Section 2.2) this is the full multibase-encoded form with no abbreviation.

actionType The tool or action identifier string (e.g., "commerce_preflight").

scopeRequired An array of scope strings required for this action

(e.g., ["commerce:read", "commerce:write"]). Each scope string MUST be in Unicode Normalization Form C (NFC). Implementations MUST sort the array by Unicode code point before canonicalization. Scope strings are compared and sorted as-is; this document does not define case folding, and scope strings that differ only in case are distinct.

timestamp An RFC 3339 UTC timestamp at exactly second precision, with the literal "Z" zone designator and no fractional-seconds component (e.g., "2026-04-08T12:00:00Z"). Implementations MUST format the timestamp to second precision before canonicalization; a timestamp carrying a fractional-seconds component is non-conforming.

The canonicalize function MUST follow RFC 8785 (JSON Canonicalization Scheme) [RFC8785]. The SHA-256 hash is computed over the UTF-8 encoding of the canonicalized JSON string.

The action_ref MUST be deterministic: two input objects that are equal field-by-field, with each field in the canonical form specified above, MUST produce the same hash regardless of implementation language, JSON library, or field insertion order. Cross-engine correlation holds only when both engines reach the computation with byte-equal canonical field values; the field-form rules above exist to make that achievable across independent implementations.

Implementations MUST NOT include additional fields in the input object. Implementations MUST NOT apply any transformation to field values beyond the NFC normalization of scopeRequired strings and the code-point sorting of the scopeRequired array specified above.

5. Signed Receipts

This section defines the signed receipt layer produced by APS policy evaluation. The policy chain of Section 4 produces a signed record at each stage; this section defines the receipt: the artifact that records what an agent declared, what a policy engine decided, and what an enforcement boundary observed. A receipt is the protocol’s unit of after-the-fact verifiability.

A receipt MUST be a signed object. A receipt MUST be content-addressed. A receipt MUST state the claims it carries in a form that a verifier can check without access to the system that produced it.

5.1. Receipt Envelope

All APS receipts share a common envelope. The envelope binds a receipt to an issuer, a point in time, and the action it concerns, independent of the receipt's type-specific payload. The envelope contains:

receipt_id The content address of the receipt (Section 5.2).

receipt_type A value identifying the receipt type. Core receipt types are defined in Section 5.3; implementations MAY define additional types as described in Section 5.3.3.

issuer The DID of the entity signing the receipt.

subject_agent The DID of the agent whose action the receipt concerns. The issuer and the subject_agent MAY differ; for a Bilateral Receipt there are two signing parties.

action_ref The content-addressed request identity defined in Section 4.1, identifying the action this receipt concerns.

delegation_ref A reference to the delegation chain under which the action was taken, or to the authority basis where no delegation applies.

decision_ref The decision identity binding the action to its evaluated authority and policy context. REQUIRED for receipts at or after the PolicyDecision stage. The concrete computation of decision_ref is deployment-profile-defined unless specified by a future APS profile; this document defines its role and binding semantics (Section 5.4), not a single normative computation.

issued_at An RFC 3339 UTC timestamp.

evidence_refs Zero or more references to artifacts the receipt relies on or records. References, not embedded content.

result The outcome or status the receipt records, in a form determined by receipt_type.

prev OPTIONAL. The content address of the preceding receipt in a chain, or null or absent for a chain origin or an unchained receipt (Section 5.2).

sig A signature over the canonicalized envelope, computed with the sig field absent.

The canonicalization of the envelope for both content-addressing and signing MUST follow RFC 8785 [RFC8785]. The signature MUST be computed as specified in Section 5.2.

A receipt MUST state what it proves. A receipt MUST NOT be interpreted as proving claims outside the semantics of its receipt_type (Section 5.3).

5.2. Content Addressing and Hash Chaining

A receipt's content address is computed as receipt_id = SHA-256(canonicalize(envelope without sig or receipt_id)), where canonicalize is RFC 8785 [RFC8785]. The receipt_id MUST be deterministic: the same envelope content MUST produce the same receipt_id regardless of implementation language or JSON library. This is the same determinism property required of action_ref in Section 4.1, applied to the receipt envelope.

Receipts MAY be chained. When chained, each receipt's prev field carries the receipt_id of the preceding receipt. A chain proves ordering between the receipts it links. A chain does not, by itself, prove completeness or prove that no receipt was omitted between two linked entries. A party that emits a chain selects which receipts the chain links; a verifier cannot conclude from the chain alone that no receipt was withheld. Deployments that require completeness or omission-resistance MUST specify an external mechanism that provides it, such as a monotonic sequence committed by the issuer or an append-only log with inclusion proofs. This document does not define such a mechanism.

The signature over the envelope MUST be an EdDSA signature as specified in [RFC8032], computed over the RFC 8785 canonicalization of the envelope with the sig field absent.

5.3. Receipt Type Categories

APS defines receipt categories at two tiers. The foundational tier covers the receipts every governed action produces. The supporting tier defines optional protocol receipt categories for deployments that need to record accountability events around an action.

5.3.1. Foundational Receipts

Action Receipt Records the occurrence and outcome of a governed action. The PolicyReceipt stage of the policy chain (Section 4) is the point at which an enforcement boundary emits an Action Receipt. Action Receipt is the general category for "this governed action occurred and here is its outcome."

Authority-Boundary Receipt Records the agent's authority position at the moment of an action: the lattice element (Section 3.1) the agent held, and which constraint dimensions bounded the action.

Completion Receipt Records the closure of a previously authorized action, and MUST reference the receipt that authorized it. A Completion Receipt closes an authorize-then-complete pair. Not every Action Receipt is a Completion Receipt: an Action Receipt becomes a Completion Receipt only when it closes a prior authorization in this way.

5.3.2. Supporting Receipts

Bilateral Receipt A receipt over an interaction between two agents, signed by both. A Bilateral Receipt proves that both parties observed the same interaction content; it does not prove either party's intent or honesty beyond that shared observation.

Custody Receipt Records a transfer of governance custody over a data object, tool, or resource from one agent to another. Custody in this sense is the responsibility for an object under the protocol's authority model; it is not ownership, possession, or any commercial-settlement relationship.

Contestability Receipt Records that an action, receipt, or decision can be challenged, or was challenged, under a defined governance process, together with the disposition of that challenge. It is not a payment dispute, an arbitration record, or a transactional-settlement artifact; those are outside the scope of this document.

5.3.3. Extension and External Representation

An implementation MAY define additional receipt types. Additional types MUST use the common envelope (Section 5.1) and MUST NOT redefine the semantics of the categories above.

APS receipts MAY be represented in external attestation formats. For example, an Action Receipt MAY be expressed as an in-toto predicate for interoperability with attestation tooling that consumes that format. Such an external envelope is a representation of an APS receipt; it does not define the APS receipt taxonomy, and the categories above remain the normative reference.

5.4. Binding to action_ref, Delegation, and Policy Decision

A receipt is bound to the action it concerns through three references.

action_ref REQUIRED. Every receipt envelope carries the action_ref of Section 4.1. Two receipts carrying the same action_ref concern the same request. This is the primary correlation anchor: it allows receipts produced by independent engines, or at different stages, to be associated without prior coordination.

delegation reference REQUIRED where the action was taken under a delegation. The receipt MUST identify the delegation chain under which the agent acted. This binds the receipt to the authority path of Section 3: a verifier can follow from the receipt to the delegation to the issuing passport.

decision reference REQUIRED for receipts at or after the PolicyDecision stage. The receipt MUST carry the decision identity of the policy evaluation it records or follows from. The action_ref is the action correlation anchor; the decision identity binds that action reference to the evaluated authority and policy context and to the decision output. It is not a second action identifier. Two receipts carrying the same decision identity claim the same action, authority and policy context, and decision output as encoded in that decision identity.

Through these three references a single receipt is locatable within the protocol: which request (action_ref), under whose authority (delegation), evaluated to what (decision identity).

5.5. Replay and Verification Expectations

The receipt signature and the canonical envelope MUST be verifiable offline. A verifier in possession of a receipt and the issuer's public key MUST be able to confirm, without network access and without contact with the issuing system: (1) that the signature is valid over the canonicalized envelope (Section 5.2); (2) that the receipt_id matches the envelope content; (3) that the action_ref is well-formed per Section 4.1; and (4) where a prev reference is present, that it is a well-formed content address.

Any claim that depends on external state, such as revocation status, policy registries, custody context, the resolution of a delegation chain, or referenced evidence artifacts, requires the corresponding referenced material and is therefore not verifiable from the receipt alone. The offline guarantee covers the receipt's own integrity, not the external facts the receipt references.

Two conforming engines evaluating the same action under the same canonical input object, authority state, and policy context MUST produce receipts that agree on action_ref. Cross-engine agreement on decision identity is defined only for conforming engines operating

under the same APS profile, where that profile specifies the same canonical decision input object and the same decision_ref computation (Section 5.4). This is the cross-engine replay property: within a shared profile, a second engine can re-evaluate a recorded decision and confirm agreement without coordinating with the first. This document does not define a globally interoperable decision_ref computation; cross-profile decision-identity agreement is therefore outside its scope.

A verifier MUST NOT treat a valid receipt signature as evidence of the truth of the receipt's claims about the external world. A receipt signature proves that the issuer attests the payload. It does not prove the payload corresponds to external fact. The distinction between protocol-level validity and external truth is addressed in the Security Considerations.

5.6. Relationship to the Policy Chain

The three-signature policy chain of Section 4, ActionIntent, PolicyDecision, PolicyReceipt, and the receipts of this section are the same structure described at two levels. The policy chain names the three signing stages of a governed action. The receipt is the signed object each stage produces and the envelope that makes it verifiable after the fact.

Specifically: the ActionIntent is the agent's signed declaration; the PolicyDecision is the policy engine's signed evaluation, carrying the decision identity of Section 5.4; the PolicyReceipt is the enforcement boundary's signed Action Receipt (Section 5.3) recording execution. PolicyReceipt names the policy-chain stage at which an Action Receipt is emitted; it is not a separate receipt_type defined by this document. Invariant INV-6 (Intent-Receipt Binding, Section 3.3) is the requirement that these are cryptographically linked: the receipt at the end of the chain MUST be traceable to the intent at its start. While general receipt chaining is optional (Section 5.2), the receipts of a single policy chain MUST carry sufficient references -- through prev, action_ref, and decision_ref -- for a verifier to trace from the final receipt back to the originating ActionIntent. INV-6 is the statement of that requirement; general chaining of unrelated receipts remains at the emitter's discretion.

The policy chain is therefore complete only with the receipt layer. Section 4 specifies when the signatures are produced; this section specifies what they produce and how it is verified. Without receipts, the policy chain defines a decision process but leaves no protocol-level artifact for later verification.

5.7. Attribution and Beneficiary References

APS receipts identify the authority, action, and evidence associated with an agent operation. Some deployments also need to attribute the resulting deliverable, or the downstream benefit of an action, to one or more entities, including humans, organizations, agents, data sources, or tool providers.

The attribution axes APS distinguishes, and the boundary of what this document specifies, are described in the Attribution Axes and Scope section. Within the receipt layer specifically, this document does not define a contribution-attribution model, a beneficiary-attribution model, or a settlement model. Implementations MAY include, in a receipt's evidence_refs (Section 5.1), references to external attribution or beneficiary records. Such records, and the allocation rules that interpret them, are expected to be specified by companion documents. A reference of this kind does not change the receipt's own semantics: a receipt continues to prove only the claim defined by its receipt_type (Section 5.3).

The principal on whose behalf an agent acted is normally resolvable from the delegation_ref carried by the receipt: the delegation chain root identifies the authorizing principal or authority basis. This document does not define a separate principal field on receipts. Deployments that need faster lookup MAY denormalize the chain root into a profile-defined field, but such a field is a convenience copy and does not replace verification of the delegation chain. Allocation of work, credit, benefit, compensation, ownership, or liability across beneficiaries or other attribution recipients remains out of scope for this document.

6. Institutional Governance

Deployments serving multi-agent organizations often need structure above the level of a single delegation chain: a charter that bounds what an organization's agents may do, offices that hold standing authority, approval policies, succession, and federation across organizations. APS is designed so that such structures compose with the delegation lattice rather than sitting beside it: each layer constrains the one below it -- a charter constrains offices, an office constrains the delegations issued under it, a delegation constrains actions -- and each containment relation is an ordering relation in the product lattice of Section 3.1, so the monotone-narrowing property holds across the full institutional structure and not only across direct delegation chains.

This document does not normatively specify the institutional structures themselves -- their wire formats, their lifecycle, or their individual semantics. It specifies only that they compose with the delegation lattice under componentwise ordering. The normative specification of institutional governance structures is companion work.

7. Protocol Bindings

7.1. MCP Binding

APS specifies a binding to MCP. APS provides an MCP server that acts as the enforcement gateway. For the purposes of this binding, a privileged action is any action the policy chain (Section 4) is configured to evaluate, including any action whose required scope is non-empty in the Scope dimension of the authority lattice (Section 3.1). In an APS-mediated MCP deployment, privileged actions covered by APS policy MUST pass through the gateway, which validates the delegation chain, evaluates the policy chain, and generates signed receipts. In deployments where the gateway is the exclusive holder of the target API credentials, the agent cannot bypass the gateway for those privileged actions.

7.2. Other Bindings

A binding to A2A exists in the reference implementation. This document specifies the MCP binding; other protocol bindings are not normatively specified here.

8. Security Considerations

The protocol’s strongest guarantees hold when all privileged effects are mediated by the ProxyGateway enforcement boundary. When agents use the SDK voluntarily without an external gateway, guarantees are conditional on agent cooperation. The threat model defines four attacker classes: adversarial agent, messaging attacker, runtime attacker, and compromised-but-signing agent. Runtime compromise is out of scope for protocol guarantees.

The compromised-but-signing agent is a hijacked agent producing cryptographically valid signatures over false claims about its own state. The agent’s signature verifies because the key is uncompromised, but the signed claims (instruction context observed, decision process attested, action taken under declared instruction X) may be false. Closure of this attack class requires operationally-independent witnesses signing claims about agent state; this is an open architectural direction (see Section 11).

This epistemic boundary extends to signed receipts. The gap between what a cryptographic governance protocol can prove and what it cannot is analyzed in [APS-EVIDENCE-GAP]. A signed receipt attests what the policy chain observed and decided. It does not attest ground truth about the external world. A valid receipt signature proves that the issuer attests the receipt's payload; it does not prove the payload corresponds to external fact. Verifiers MUST treat receipts as evidence of what was attested, not as proof of what is true.

9. IANA Considerations

This document has no IANA actions.

10. Attribution Axes and Scope

APS distinguishes four attribution axes for agentic work. They are distinct because, in multi-party workflows, they may resolve to different entities.

Authority The delegation chain under which an action was permitted. This document specifies the authority axis in full (Sections 3 and 4).

Contribution The sources, such as data, models, tools, or prior work product, that fed into a deliverable. This document does not specify a contribution-attribution model.

Principal The entity on whose behalf, or under whose account, the agent acted. The principal is resolvable from the delegation chain root referenced by the receipt envelope (see Section 5.7).

Beneficiary The entity that receives the value, credit, or downstream benefit the work creates. This document does not specify a beneficiary-attribution model.

The distinction between the principal and the beneficiary is structurally load-bearing. An agent may act on behalf of an authorizing principal, such as a consulting firm, to produce a deliverable whose beneficiary is a third party, such as that firm's client. A protocol that collapses the two axes cannot represent this ordinary case.

The authority and principal axes are resolvable from the mechanisms this document defines when the referenced delegation material is available. The contribution and beneficiary axes are recognized as first-class attribution layers of APS whose normative mechanics are left to companion specifications. The allocation of credit, benefit, compensation, liability, or ownership across beneficiaries is out of scope for this document and is not implied by the authority chain.

11. Future Work

Future work includes wire-format epistemic discipline: typed claim and evidence labels that make explicit what each signed artifact proves and what it does not prove. Claim and evidence typing exist in the reference implementation; standardizing those labels as a normative wire-format is left to future work.

Several elements named in this document are deferred to companion specifications: a normative decision_ref computation for cross-profile interoperability (Section 5.4); the contribution-attribution and beneficiary-attribution layers (Attribution Axes and Scope); the normative specification of institutional governance structures (Section 6); and operationally-independent witnesses for the compromised-but-signing agent class (Section 8).

12. References

12.1. Normative References

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

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

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

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

12.2. Informative References

[APS-NARROWING]
Pidlisnyi, T., "Monotonic Narrowing for Agent Authority", March 2026, <https://doi.org/10.5281/zenodo.18932404>.

[APS-FACETED]
Pidlisnyi, T., "Faceted Authority Attenuation", March 2026, <https://doi.org/10.5281/zenodo.19260073>.

[APS-EVIDENCE-GAP]
Pidlisnyi, T., "The Evidence-Safety Gap in Cryptographic Agent Governance", April 2026, <https://doi.org/10.5281/zenodo.19476002>.

Appendix A. Implementation Status

Reference implementations of APS are available in TypeScript and Python, published as open-source packages under Apache-2.0: the TypeScript SDK and the Python SDK are both published as "agent-passport-system" on npm and PyPI respectively, and an MCP server binding is published as "agent-passport-system-mcp" on npm. The implementations carry a conformance test suite and a published conformance fixture set. Current versions, test counts, tool counts, and conformance results are maintained at the project repository: https://github.com/aeoess/agent-passport-system.

Author’s Address

Tymofii Pidlisnyi
AEOESS
Email: signal@aeoess.com
URI: https://aeoess.com