Individual Submission
Internet-Draft
Intended status: Informational
Expires: 28 September 2026

S. Prakash
Independent

27 March 2026

Agent Identity Protocol (AIP): Verifiable Delegation for AI Agent Systems
draft-prakash-aip-00

Abstract

This document specifies the Agent Identity Protocol (AIP), a protocol for verifiable, delegable identity for AI agent systems. AIP introduces Invocation-Bound Capability Tokens (IBCTs) that bind identity, authorization, scope constraints, and provenance into a single cryptographic artifact. Two token modes are defined: a compact mode using JSON Web Tokens (JWT) with Ed25519 signatures for single-hop interactions, and a chained mode using Biscuit tokens with append-only blocks and Datalog policy evaluation for multi-hop delegation chains. Protocol bindings are specified for the Model Context Protocol (MCP), Agent-to-Agent Protocol (A2A), and generic HTTP APIs. The protocol addresses authentication gaps in current AI agent infrastructure where a survey of approximately 2,000 MCP servers found all lacked authentication.

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 28 September 2026.

Copyright Notice

Table of Contents

1. Introduction

AI agent systems are increasingly deployed in multi-agent architectures where an orchestrator decomposes tasks and delegates subtasks to specialist agents. The protocols enabling this communication, notably the Model Context Protocol (MCP) and the Agent-to-Agent Protocol (A2A), solve the connectivity problem but do not solve the identity problem.

MCP provides no built-in authentication layer. A2A uses self-declared identities with no attestation mechanism. OAuth 2.1, recently added to MCP, covers single-hop client-to-server authentication but does not address multi-hop delegation chains. When an orchestrator delegates to a specialist that calls a tool, the delegation chain that led to the tool invocation is lost.

AIP fills this gap by introducing Invocation-Bound Capability Tokens (IBCTs) that answer four questions for every agent action: who authorized this action, through which delegation chain, with what constraints at each hop, and what was the outcome.

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

AIP defines two identifier schemes for agent identity:

2.1. DNS-Based Identifiers

DNS-based identifiers follow the format:

aip:web:<domain>/<path>

Example: aip:web:example.com/agents/research-analyst

DNS-based identifiers are suitable for long-lived agents with stable domain ownership. Identity documents are resolved via HTTPS at a well-known path.

2.2. Self-Certifying Identifiers

aip:key:ed25519:<multibase-encoded-public-key>

Self-certifying identifiers derive identity from the public key itself. They are suitable for ephemeral agents that do not require DNS infrastructure. The identifier is deterministically computed from the Ed25519 public key using multibase encoding.

2.3. Identity Document

Each agent with a DNS-based identifier MUST publish an identity document at:

https://<domain>/.well-known/aip/<path>.json

The identity document is a JSON object containing:

* aip: Protocol version (MUST be "1.0")
* id: The agent’s AIP identifier
* public_keys: Array of public key objects with validity windows
* name: Human-readable agent name
* delegation: Delegation preferences
* protocols: Supported protocol bindings
* document_signature: Ed25519 signature over the canonicalized document
* expires: Document expiration timestamp

The document MUST be self-signed. Verification uses JSON Canonicalization Scheme (JCS) [RFC8785]: remove the document_signature field, canonicalize the remaining JSON, and verify the Ed25519 signature against a currently-valid public key.

3. Token Formats

AIP defines two token modes that share a common identity scheme but differ in delegation capability.

3.1. Compact Mode (JWT)

Compact mode tokens are JSON Web Tokens [RFC7519] signed with Ed25519 (EdDSA). They support single-hop interactions only.

Header:

{"alg": "EdDSA", "typ": "aip+jwt"}

Claims:

* iss: Issuer AIP identifier (REQUIRED)
* sub: Subject/holder AIP identifier (REQUIRED)
* scope: Array of authorized capabilities (REQUIRED)
* budget_usd: Authorization budget ceiling in USD (REQUIRED)
* max_depth: Maximum delegation depth, 0 for no further delegation (REQUIRED)
* iat: Issued-at timestamp (REQUIRED)
* exp: Expiration timestamp (REQUIRED)

Token lifetime SHOULD be less than one hour. Both iss and sub MUST be valid AIP identifiers.

3.2. Chained Mode (Biscuit)

Chained mode tokens use Biscuit [BISCUIT] tokens with append-only blocks and Datalog policy evaluation. They support multi-hop delegation with cryptographic scope attenuation.

A chained token consists of ordered blocks:

* Block 0 (Authority): Root identity, initial capabilities, budget, max_depth, expiration. Signed by the root authority.
* Blocks 1..N-1 (Delegation): Each block narrows scope. Contains delegator, delegate, attenuated capabilities (as Biscuit right facts), attenuated budget, and a non-empty context field. Signed by the delegator.
* Block N (Completion, optional): Execution outcome. Contains status, result_hash (SHA-256), verification_status, and optional resource usage metrics. Signed by the executing agent.

3.3. Scope Attenuation

Scope attenuation is a fundamental security property of AIP. At each delegation step, scope can only narrow or remain equal, never widen. This applies across four dimensions:

* Tools: Child scope MUST be a subset of parent scope
* Budget: Child budget MUST be less than or equal to parent budget
* Domains: Child domains MUST be a subset of parent domains
* Time: Child expiration MUST be less than or equal to parent expiration

Verifiers MUST check attenuation at every hop in the delegation chain. A wildcard (*) in the parent permits any specific value in the child, but a specific value in the parent MUST NOT widen to a wildcard in the child.

3.4. Policy Profiles

Chained mode tokens support three policy profiles of increasing expressiveness:

* Simple: Templated rules requiring no Datalog knowledge. The library generates canonical Datalog for tool allowlists, budget ceilings, delegation depth, and time expiry.
* Standard: Curated Datalog subset without recursion and with bounded evaluation.
* Advanced: Full Datalog with a maximum 1000 iteration limit. Opt-in only.

3.5. Budget Semantics

Budget values in AIP tokens represent per-token authorization ceilings, NOT running balances. A delegator asserts "I authorize up to $X for this task" at delegation time. The verifier checks that the budget is non-negative but does NOT track cumulative spending. Aggregate budget enforcement is the responsibility of the orchestration platform at runtime. Completion blocks record actual cost_usd for audit purposes.

4. Protocol Bindings

4.1. MCP Binding

AIP tokens are transported in MCP via the X-AIP-Token HTTP header:

X-AIP-Token: <compact-or-chained-token>

For tokens exceeding 4KB, token-by-reference is supported:

X-AIP-Token-Ref: https://issuer/.well-known/aip/tokens/<id>

MCP servers verify tokens in five steps: (1) extract token, (2) verify signatures against issuer identity document, (3) check requested tool against token scope, (4) validate chain constraints for chained tokens, (5) inject verified identity into request context.

Nine error codes are defined with appropriate HTTP status mappings: 401 for authentication failures (token_missing, token_malformed, signature_invalid, identity_unresolvable, token_expired, key_revoked) and 403 for authorization failures (scope_insufficient, budget_exceeded, depth_exceeded).

Servers declare AIP requirements via the require_aip field in their identity document's protocols.mcp configuration.

4.2. A2A Binding

In A2A interactions, AIP tokens are transported in the metadata.aip_token field of task submissions. Agent cards are extended with an aip_identity object containing the agent's AIP identifier and document URL.

The A2A verification flow adds a sixth step: the calling agent appends a delegation block with attenuated scope before sending the task, and the receiving agent verifies that the final delegation block delegates to its own AIP identifier.

4.3. HTTP Binding

For generic HTTP APIs not using MCP or A2A, tokens are transported via the Authorization header with the AIP scheme:

Authorization: AIP <base64url-encoded-token>

Token-by-reference uses the X-AIP-Token-Ref header with a 5-second fetch timeout and SSRF protection (reject reference URLs outside expected domain patterns).

5. Delegation Lifecycle

5.1. Bounded Depth

Block 0 declares max_depth (default: 3). Each delegation block increments effective depth by 1. If current depth equals max_depth, further delegation is forbidden. In compact mode, max_depth of 0 means the holder MUST NOT delegate further.

5.2. Delegation Context

Each delegation block MUST include a non-empty context field containing a human-readable description of the delegation purpose. Verifiers MUST reject tokens with missing or empty context. This requirement ensures audit trail integrity.

5.3. Ephemeral Agent Grants

For short-lived sub-agents, a parent agent generates an Ed25519 keypair, creates an aip:key: identifier, and appends a delegation block with scoped capabilities and a short TTL (5 minutes RECOMMENDED). The parent’s identity document MAY set delegation.allow_ephemeral_grants to false to prevent this.

5.4. Key Rotation

DNS-based identifiers support zero-downtime key rotation through overlapping validity windows on public keys. A new key is published with a future valid_from timestamp. Both keys are valid during the overlap period. Recommended rotation period is 90 days. Cache TTL MUST NOT exceed 5 minutes.

Self-certifying identifiers cannot rotate keys; key rotation requires identity replacement, which is acceptable for ephemeral agents.

5.5. Revocation

AIP prefers short-lived tokens over revocation infrastructure. Compact mode tokens SHOULD have a TTL under 1 hour, making revocation generally unnecessary. For chained mode, key revocation (removing a key from the identity document) invalidates all tokens signed by that key. Token-specific revocation via Certificate Revocation Lists is deferred to v2.

6. Provenance and Audit

6.1. Completion Blocks

A completion block is the final block in a chained token, signed by the executing agent. It contains:

* status: REQUIRED. One of "completed", "failed", or "partial".

* result_hash: REQUIRED. SHA-256 hash of the output in format "sha256:<hex>".

* verification_status: REQUIRED. One of "self_reported", "tool_verified", "peer_verified", or "human_verified".

* tokens_used: OPTIONAL. LLM tokens consumed.

* cost_usd: OPTIONAL. Actual cost incurred.

* duration_ms: OPTIONAL. Wall-clock execution time.

* ldp_provenance_id: OPTIONAL. Back-link to LDP provenance record.

6.2. Verification Trust Levels

AIP defines three escalating trust levels for completion data:

* Level 1 (Self-Reported): Agent reports its own results with no independent verification. Default for trusted environments.

* Level 2 (Counter-Signed): Delegator independently verifies the result and appends a verification block.

* Level 3 (Third-Party Attested): External verifier (LDP peer, human reviewer, or audit service) signs an attestation block.

6.3. Audit Tokens

A completed chained token with a completion block appended is a self-contained audit artifact. It answers five questions without requiring an external database: who authorized (Block 0), through whom (delegation blocks), what constraints (Datalog policies), what happened (completion block), and whether it was verified (verification_status). Audit tokens are tamper-evident, non-repudiable, and verifiable offline using public keys from identity documents.

7. Security Considerations

This section addresses the security properties and threat model for AIP.

7.1. Threat Model

AIP is designed to resist the following attack categories:

* Scope widening: An agent attempts to exceed its delegated capabilities. Prevented by cryptographic scope attenuation verification at each hop.

* Delegation depth violation: An agent attempts to delegate beyond the maximum permitted depth. Prevented by depth tracking in each delegation block.

* Token replay: A captured token is reused. Mitigated by short TTLs (under 1 hour recommended for compact mode).

* Token forgery: An attacker constructs a token without holding the private key. Prevented by Ed25519 signature verification.

* Identity spoofing: An agent claims a false identity. Prevented by identity document resolution and signature verification.

* Audit evasion: An agent delegates with empty context to avoid audit trails. Prevented by mandatory non-empty context on all delegation blocks.

7.2. Adversarial Evaluation

Experimental evaluation across 600 adversarial attempts in six attack categories showed a 100% rejection rate. Two attack categories (delegation depth violation and audit evasion through empty context) are uniquely addressed by AIP’s chained token structure and cannot be detected by standard JWT deployments. Details are reported in the companion paper [AIP-PAPER].

7.3. Cryptographic Agility

AIP v1 mandates Ed25519 exclusively. No algorithm negotiation is supported. This is a deliberate design choice to eliminate downgrade attacks and reduce implementation complexity. Future versions MAY introduce additional algorithms through the protocol version field.

7.4. Transport Security

Identity document resolution and token-by-reference fetching MUST use HTTPS. Implementations SHOULD enforce TLS 1.3 or later. Token-by-reference URLs MUST be validated against expected domain patterns to prevent SSRF attacks. Fetch timeout SHOULD be 5 seconds.

8. IANA Considerations

This document requests the following IANA registrations:

8.1. HTTP Authentication Scheme

Registration of the "AIP" HTTP authentication scheme in the "HTTP Authentication Scheme Registry":

* Authentication Scheme Name: AIP
* Reference: This document, Section 4.3

8.2. Well-Known URI

Registration of the "aip" well-known URI suffix in the "Well-Known URIs" registry:

* URI Suffix: aip
* Change Controller: IETF
* Reference: This document, Section 2.3

8.3. Media Type

Registration of the "aip+jwt" structured syntax suffix:

* Type name: application
* Subtype name: aip+jwt
* Reference: This document, Section 3.1

9. References

9.1. 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>.

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

[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>.

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

9.2. Informative References

[BISCUIT] Music, G., "Biscuit Authorization Token", 2024, <https://www.biscuitsec.org/>.

[AIP-PAPER] Prakash, S., "AIP: Agent Identity Protocol for Verifiable Delegation Across MCP and A2A", 27 March 2026, <https://arxiv.org/abs/2603.24775>.

Acknowledgements

The Biscuit authorization token specification influenced the chained mode design. The MCP and A2A protocol teams provided the agent communication infrastructure that AIP extends.

Author’s Address

Sunil Prakash
Independent
Email: sunil@sunilprakash.com
URI: https://sunilprakash.com