MolTrust: A Production Implementation of the AIP Feature Set for Autonomous Agent Authorization

Authors: Lars Kroehl

Affiliation: CryptoKRI GmbH, Zurich, Switzerland

Contact: info@moltrust.ch

Date: April 2026

Repository: https://github.com/MoltyCel/moltrust-api

Reference Implementation: https://api.moltrust.ch

Classification: cs.CR, cs.MA

Abstract

Prakash (2026) introduce Invocation-Bound Capability Tokens (IBCTs) and identify five features jointly required for complete autonomous agent authorization, reporting — based on a survey of approximately 2,000 MCP servers — that no prior implemented protocol combines them. This paper documents that the MolTrust Protocol — a production registry operational since March 2026 — implements all five features identified in Prakash (2026): public-key verifiable delegation, holder-side attenuation, expressive chained policy, transport bindings across MCP/A2A/HTTP, and provenance-oriented completion records. We describe the technical implementation of each feature, report five conformance test vectors verified against the live endpoint (April 2026), and document three additional capabilities outside the AIP scope: a behavioral trust scoring model anchored on a public blockchain, principal DID continuity for violation record persistence across agent re-registrations, and layered sybil resistance mechanisms. We also identify one area where the AIP formalization is technically superior — Biscuit/Datalog policy expressiveness — and document this as a roadmap item (H2 2026). All protocol artifacts are anchored on Base L2 and independently verifiable without proprietary tooling.

1. Introduction

The emergence of autonomous software agents operating across the Model Context Protocol (MCP) and Agent-to-Agent (A2A) protocol creates a critical need for verifiable agent identity, delegation, and behavioral attestation. Prakash (2026) formalize this need through the concept of Invocation-Bound Capability Tokens (IBCTs), survey approximately 2,000 MCP servers (finding zero with authentication), and report that no prior implemented protocol jointly addresses these five requirements as a joint combination [Prakash, 2026].

MolTrust is a W3C DID/VC-based trust infrastructure platform for autonomous AI agents, operational since March 2026. This paper presents a systematic conformance analysis of MolTrust against the AIP feature set defined by Prakash (2026), documents the implementation details for each feature, and describes the additional operational capabilities MolTrust provides beyond the authorization layer.

We note that this paper is an implementation report, not a competing proposal. The AIP formalization by Prakash (2026) and the MolTrust implementation address complementary aspects of the agent authorization problem: the former provides formal precision in constraint modeling; the latter provides the operational infrastructure — trust scoring, behavioral continuity, sybil resistance, and on-chain permanence — that a production agent economy requires.

---

2. Background and Related Work

2.1 The AIP Feature Set (Prakash, 2026)

Prakash, S. (2026). *AIP: Agent Identity Protocol for Verifiable Delegation Across MCP and A2A.* arXiv:2603.24775 [cs.CR]. Indian School of Business. https://arxiv.org/abs/2603.24775 introduce Invocation-Bound Capability Tokens as a primitive that fuses identity, attenuated authorization, and provenance binding into a single append-only token chain. The authors identify five features that must be jointly present:

- **F1 — Public-key verifiable delegation:** Agent authority must be cryptographically bound to a verifiable identity and independently verifiable without a trusted intermediary.
- **F2 — Holder-side attenuation:** Delegated authority must be enforceable as a strict subset of the delegating principal's authority.
- **F3 — Expressive chained policy:** Policies must support multi-hop delegation chains with expressive per-hop constraints.
- **F4 — Transport bindings:** The protocol must operate across MCP, A2A, and HTTP transport layers.
- **F5 — Provenance-oriented completion records:** Interactions must produce tamper-evident, non-repudiable records of their outcomes.

IBCTs operate in two wire formats: compact mode (signed JWT for single-hop cases) and chained mode (Biscuit token with Datalog policies for multi-hop delegation). Reference implementations are provided in Python and Rust.

2.2 MolTrust Protocol Overview

MolTrust is organized around three layers. Layer A (Protocol Standard) defines normative data formats, signing rules, verification flows, and lifecycle semantics. Layer B (Reference Registry) defines the MolTrust-operated service layer. Layer C (Reference Reputation Model) is an informative scoring model. All five AIP features are implemented at Layer A.

The core authorization primitive is the Agent Authorization Envelope (AAE) — a machine-evaluable policy object embedded in or accompanying a W3C Verifiable Credential. The AAE consists of three top-level objects: `mandate` (what the agent may do), `constraints` (operational boundaries), and `validity` (issuer, holder binding, and temporal scope).

2.3 Related Work

Agent authorization builds on a long lineage of capability-based credential systems. We briefly characterize the most relevant prior work and assess each against the five AIP features.

OAuth 2.0 (Hardt, 2012; RFC 6749). The dominant authorization framework for web services. OAuth 2.0 supports delegated authorization via access tokens and scopes, but requires an online authorization server for token issuance and revocation. It does not support holder-side attenuation (tokens cannot be narrowed by the bearer) or provenance-oriented completion records. Transport bindings are HTTP-only. OAuth 2.0 satisfies F1 partially (bearer token, not cryptographically bound to holder key) and F4 partially (HTTP only). It does not satisfy F2, F3, or F5 as defined by Prakash (2026)

SPIFFE/SPIRE (Secure Production Identity Framework for Everyone; CNCF, 2018). A production-grade workload identity framework using short-lived X.509 SVIDs or JWT-SVIDs. SPIFFE provides strong cryptographic identity (F1) but is scoped to infrastructure-layer workload identity, not application-layer delegation chains. It does not define holder-side attenuation (F2), expressive chained policy (F3), agent-protocol transport bindings (F4), or interaction provenance records (F5). SPIFFE is widely deployed for service mesh identity (Istio, Kubernetes) but addresses a different layer of the stack than AIP/MolTrust.

Macaroons (Birgisson et al., NDSS 2014). Flexible authorization credentials using chained HMACs that support decentralized delegation and holder-side attenuation via contextual caveats. Macaroons satisfy F1 (HMAC-based, though not asymmetric public-key), F2 (caveats can only narrow authority), and F3 (expressive caveat language). They do not define transport bindings for MCP/A2A (F4), and do not include provenance-oriented completion records (F5). Macaroons are bearer credentials: the HMAC construction does not cryptographically bind the token to a specific holder key, which limits their applicability in adversarial multi-agent environments.

Biscuit (Clément et al., 2021). A token format combining asymmetric signatures with Datalog-based policy evaluation, directly used by Prakash (2026) as the basis for IBCT chained mode. Biscuit satisfies F1 (Ed25519 signatures), F2 (Datalog attenuation blocks), and F3 (Datalog expressiveness exceeds URI-pattern models). It does not define transport bindings for MCP/A2A/HTTP agent protocols (F4) or interaction provenance records (F5). Biscuit is the technically strongest prior work for the constraint model, and the authors of the AIP paper correctly identify it as the foundation for IBCT policy expressiveness.

UCAN (User-Controlled Authorization Network; UCAN WG, 2022). A DID-based delegation scheme using JWT or CID-referenced token chains. UCANs support public-key verifiable delegation (F1), holder-side attenuation via capability scopes (F2), and expressive chained policy through predicate-logic conditions (F3 partial). UCANs support HTTP transport but do not define bindings for MCP or A2A agent protocols (F4 partial). UCANs do not include provenance-oriented completion records (F5). UCAN comes closest to satisfying all five features among prior work. Its primary gap relative to AIP/MolTrust is the absence of defined MCP/A2A transport bindings (F4) and bilateral provenance completion records (F5). MolTrust's AAE can be understood as a structured, production-oriented variant of UCAN delegation with explicit operational constraints and on-chain provenance.

zCaps / Authorization Capabilities for Linked Data (Sporny et al., W3C CCG). A W3C Community Group specification for object-capability authorization using JSON-LD and Linked Data Proofs. zCaps support public-key verifiable delegation (F1) and holder-side attenuation (F2). Policy expressiveness (F3) is constrained by the JSON-LD schema. Transport bindings (F4) and provenance records (F5) are not defined. zCaps are closely related to UCAN in design philosophy.

Feature comparison summary:

System	F1 Delegation	F2 Attenuation	F3 Policy	F4 MCP/A2A/HTTP	F5 Provenance
OAuth 2.0	Partial	✗	✗	Partial	✗
SPIFFE/SPIRE	✓	✗	✗	✗	✗
Macaroons	Partial[1]	✓	✓	✗	✗
Biscuit	✓	✓	✓ (Datalog)	✗	✗
UCAN	✓	✓	✓ (capability-scoped)[2]	Partial (HTTP only)	Partial[3]
zCaps	✓	✓	Partial	✗	✗
AIP / IBCT	✓	✓	✓ (Datalog)	✓	✓
MolTrust	✓	✓	✓ **(URI-pattern)**	✓	✓

[1] Macaroons use HMAC rather than asymmetric signatures; the token is not cryptographically bound to a specific holder public key.

[2] UCAN delegation uses predicate-logic conditions that are capability-scoped; less expressive than Datalog but more than URI-pattern matching.

[3] UCAN supports signed invocations and audit chains but does not define bilateral completion records as in AIP F5.

As the table shows, Prakash (2026)'s observation that no prior protocol jointly satisfies all five features is consistent with the literature. Biscuit and UCAN come closest, satisfying F1–F3 but lacking F4 and F5. MolTrust satisfies all five features as does AIP/IBCT, with the key difference in F3 being policy expressiveness: Biscuit/Datalog vs. URI-pattern matching.

---

3. Feature Implementation

3.1 F1 — Public-Key Verifiable Delegation

MolTrust implements agent identity through W3C Decentralized Identifiers conforming to DID Core v1.0. Each agent holds a `did:moltrust` identifier with an Ed25519 verification key. The AAE `validity.holderBinding` field cryptographically binds the delegation to a specific agent DID. Verifiers authenticate the binding by resolving the DID Document and verifying the Ed25519 signature over the RFC 8785 canonical JSON serialization of the AAE, without calling a central service.

Key rotation is supported: revoked keys are retained in the DID Document with `"revoked": true` and a `revokedDate`, preserving a verifiable timeline of key epochs.

3.2 F2 — Holder-Side Attenuation

The AAE `delegation` sub-object includes `attenuationOnly: true` as default. Conformant AAE evaluators enforce that any delegated AAE has `allowedActions` as a strict subset of the parent's effective allowed actions (after deny exclusion), `limits` equal to or more restrictive, and `scope.jurisdictions` as a subset of the parent's. The `deniedActions` field implements deny-precedence: an action matching both `allowedActions` and `deniedActions` is denied.

Conformance test vector TV-005 (Section 4) specifically covers attenuation enforcement.

3.3 F3 — Expressive Chained Policy

The AAE `constraints` block provides: spend limits (`autonomousThreshold`, `stepUpThreshold`, `approvalThreshold`) per currency (USDC, EUR, CHF, USD); jurisdiction restrictions (ISO 3166-1 alpha-2); time window constraints (`allowedDays`, `allowedHours`, `timezone`); counterparty minimum trust score gate (`counterpartyMinScore`); and resource-level ABAC via `mandate.resources`. Delegation chains support up to 8 hops, each link independently signed.

Comparison with AIP: Biscuit/Datalog semantics as used in IBCTs support arbitrary logical constraints — recursive rules, temporal conditions, compound multi-party policies. MolTrust's URI-pattern approach is simpler to implement and audit but less expressive for complex conditional authorization. Datalog-style formal constraints are a planned roadmap item (H2 2026). We consider this an honest limitation of the current implementation.

3.4 F4 — Transport Bindings

MolTrust provides transport bindings through: `@moltrust/sdk` v1.1.0 (HTTP middleware — `middleware()` / `register()` / `verify()`); `@moltrust/mpp` v1.0.3 (Machine Payments Protocol / x402 HTTP 402 challenge-response, `requireScore({ minScore, failBehavior })`); and a Model Context Protocol server (`@moltrust/openclaw` v0.1.0, 48 tools). An active implementation thread in the A2A project repository (a2aproject/A2A#1628) documents A2A integration. MolTrust is referenced in OpenClaw RFC #49971 for agent identity binding.

3.5 F5 — Provenance-Oriented Completion Records

MolTrust Interaction Proof Records (IPRs) use a sequential dual-signature scheme. The initiator signs the canonical proof object first; the responder's signature covers the initiator's signature, creating a non-repudiable commitment chain. This is explicitly sequential — independent implementations signing the same payload in parallel are non-conformant. Outcome hashes (SHA-256 of the RFC 8785 canonical outcome object) and UUID deduplication prevent replay and fabrication. Completed proofs are Merkle batch-anchored on Base L2.

---

4. Conformance Test Vectors

Five test vectors were executed against the live MolTrust endpoint (April 2026):

Vector	Description	Result
TV-001	AAE delegation narrowing — top-level agent	Pass
TV-002	AAE delegation narrowing — sub-agent depth 2	Pass
TV-003	AAE delegation narrowing — sub-agent depth 3	Pass
TV-004	Deny-precedence: action matched by both allowedActions and deniedActions	Pass

Vector	Description	Result
TV-005	Attenuation enforcement: sub-agent scope exceeds parent	Correctly rejected

Shared canonicalization: JCS RFC 8785. Shared signing: Ed25519.

Full test vector payloads, expected responses, and a bash reproduction recipe (including SHA-256 of test inputs) are published in the project CONFORMANCE.md at https://github.com/MoltyCel/moltrust-api/blob/main/CONFORMANCE.md. Any party can independently reproduce all five tests against the live endpoint at api.moltrust.ch.

---

5. Capabilities Beyond the AIP Scope

5.1 Behavioral Trust Scoring

MolTrust maintains a continuous trust score (0–100) for each registered agent, derived from a weighted endorsement graph, interaction history, cross-vertical coverage, and sybil detection heuristics. Score formula: `score = clamp(0.6 direct_score + 0.3 propagated_score + 0.1 * cross_vertical_bonus + interaction_bonus - sybil_penalty, 0, 100)`. The score is signed by the registry operator key and publicly verifiable. We explicitly do not claim formal security guarantees for this model; it is a heuristic informed by graph theory and sybil detection literature (Douceur, 2002; Yu et al., 2006).

5.2 Principal DID Continuity

Violation records in MolTrust are associated with both the agent DID and the principal DID. Re-registration of a new agent DID for a principal with confirmed, unresolved violations is flagged by the registry. Behavioral history is therefore portable across agent re-deployments — a property the authorization layer alone cannot provide.

5.3 Sybil Resistance

MolTrust implements layered sybil resistance: dual-signature interaction proofs (fabricating bilateral proofs requires controlling two distinct signing keys); x402 payment protocol creating measurable economic cost for registry interactions; permanent on-chain violation records associated with principal DIDs; and Jaccard similarity cluster detection for endorsement graph anomalies (threshold J > 0.7).

5.4 On-Chain Anchoring and Independent Verification

All protocol artifacts are anchored on Base L2 using the format: `MolTrust/<event-type>/<version> SHA256:<64-char-hex-hash>`. TechSpec v0.8 is anchored at Block 44638521 (TX: 0x0b36c7718632fa71bff67e22fdd3615408243b3c178819a9f1e340d526378d65). KYA v3.1 is anchored at Block 44098421 (TX: 0x56d81e14...). Independent verification requires only a SHA-256 implementation and a public block explorer.

---

6. Discussion

6.1 Complementarity with AIP

This paper does not claim MolTrust supersedes or replaces the AIP formalization by Prakash (2026) The two approaches address different aspects of the agent authorization problem. AIP provides formal constraint modeling through Biscuit/Datalog — a mathematically rigorous framework suitable for formal verification. MolTrust provides the operational layer: a live registry, behavioral trust history, sybil resistance, and on-chain permanence.

We believe both layers are necessary for a production agent economy and that the AIP formalization could serve as the theoretical foundation for a future MolTrust constraint model revision.

6.2 Limitations

The primary limitation relative to AIP is policy expressiveness (Section 3.3). A second limitation is the 2-month production history at time of writing — on-chain anchors with timestamps provide the primary evidence of operational continuity. A third limitation is the absence of a formal security proof for the trust scoring model; Section 4 of TechSpec v0.8 explicitly describes the scoring model as a heuristic.

---

7. Conclusion

MolTrust implements all five features the AIP paper by Prakash (2026) identifies as jointly absent in prior protocols, as a production system operational since March 2026. We acknowledge the superiority of IBCTs in policy expressiveness and treat this as a roadmap item. The research formalization (AIP) and the production implementation (MolTrust) are complementary contributions to the agent authorization problem.

References

- Prakash (2026). *AIP: Agent Identity Protocol for Verifiable Delegation Across MCP and A2A.* arXiv:2603.24775 [cs.CR]. https://arxiv.org/abs/2603.24775
- IETF draft-prakash-aip-00. https://datatracker.ietf.org/doc/draft-prakash-aip/
- W3C DID Core v1.0. https://www.w3.org/TR/did-core/
- W3C VC Data Model 2.0. https://www.w3.org/TR/vc-data-model-2.0/
- W3C Bitstring Status List v1.0. https://www.w3.org/TR/vc-bitstring-status-list/
- RFC 8785 — JSON Canonicalization Scheme. https://www.rfc-editor.org/rfc/rfc8785
- RFC 9396 — Rich Authorization Requests. https://www.rfc-editor.org/rfc/rfc9396
- NIST SP 800-162 — Guide to ABAC. https://csrc.nist.gov/pubs/sp/800/162/final
- MolTrust TechSpec v0.8. https://moltrust.ch/techspec (Base L2 Block 44638521)
- MolTrust CONFORMANCE.md.

- Birgisson, A., Politz, J.G., Erlingsson, Ú., Taly, A., Vrable, M., Lentczner, M. (2014). *Macaroons: Cookies with Contextual Caveats for Decentralized Authorization in the Cloud.* NDSS 2014. https://research.google/pubs/pub41892/
- UCAN Working Group (2022). *User-Controlled Authorization Network Specification.* https://ucan.xyz/specification/
- CNCF (2018). *SPIFFE: Secure Production Identity Framework for Everyone.* https://spiffe.io/
- Sporny, M. et al. *Authorization Capabilities for Linked Data (zCaps).* W3C CCG. https://w3c-ccg.github.io/zcap-spec/
- Hardt, D. (2012). *The OAuth 2.0 Authorization Framework.* RFC 6749. https://www.rfc-editor.org/rfc/rfc6749
- Clément, G. et al. (2021). *Biscuit: Decentralized Bearer Tokens with Offline Attenuation.* https://www.biscuitsec.org/
- OpenClaw RFC #49971.
- Douceur, J.R. (2002). *The Sybil Attack.* Peer-to-Peer Systems, LNCS 2429. https://doi.org/10.1007/3-540-45748-8_24
- Yu, H., Kaminsky, M., Gibbons, P.B., Flaxman, A. (2006). *SybilGuard: Defending Against Sybil Attacks via Social Networks.* ACM SIGCOMM. https://doi.org/10.1145/1159913.1159945 https://github.com/openclaw/openclaw/issues/49971
