IBA Working Group
Internet-Draft: IBA-SPEC-001
Intended Status: Standards Track
Expires: September 2026

J. Williams
IntentBound / GoverningLayer
March 2026
GB2603013.0 (pending)

IBA-SPEC-001 · DRAFT v0.1 · MARCH 2026

Intent-Based Authorization Protocol

IBA Protocol Specification · Version 0.1

PATENT APPLICATION	NIST FILINGS	NCCoE FILINGS	ENFORCEMENT LATENCY
GB2603013.0	13	6	<5ms
Pending · Filed 5 Feb 2026	NIST-2025-0035 · Closed	AI Agent Identity · Apr 2026	O(1) · Deterministic

This document specifies the Intent-Based Authorization (IBA) Protocol — a cryptographic runtime enforcement framework for AI agent authorization. IBA defines the format and validation rules for Intent Certificates: signed declarations that an AI agent presents to any system before executing actions on behalf of a human principal.

Status: Working Draft · Not for production use · Published for technical review and standards body consideration. Patent Application GB2603013.0 pending. RAND licensing terms intended upon patent grant. NIST-2025-0035 · 13 filings · NCCoE AI Agent Identity · 6 filings including supplemental filing re: Amazon v. Perplexity (N.D. Cal., March 10, 2026).

RFC 2119

The key words **MUST**, **MUST NOT**, **SHOULD**, **SHOULD NOT**, and **MAY** in this document are to be interpreted as described in RFC 2119 (Bradner, 1997). Implementations claiming IBA compliance are bound by all MUST and MUST NOT requirements.

1. Introduction and Motivation

1.1 The Authorization Gap

AI agents operating on behalf of human principals face a fundamental authorization problem: existing authentication systems establish identity but not intent. An agent may be verifiably authenticated as belonging to a specific user while simultaneously operating outside the scope of what that user actually authorized.

This gap has been confirmed in federal jurisprudence. In Amazon.com Services LLC v. Perplexity AI, Inc. (N.D. Cal., March 10, 2026), the court established that user consent alone is insufficient for agent platform access under the Computer Fraud and Abuse Act (CFAA). The ruling implies a technical requirement that no existing standard satisfies: a machine-readable credential specifying who the agent is, who authorized it, what it is permitted to do, and what it is forbidden from doing — presented and verifiable before any system access occurs. IBA provides that credential.

1.2 Design Goals

The IBA Protocol is designed to satisfy the following requirements:

1	**Cryptographically bind AI agent actions to verified human intent** ECDSA-P384 signature over full certificate payload; principal co-signs every certificate
2	**Provide deterministic runtime enforcement** O(1) TBDE validation <5ms; DENY_ALL default posture; no probabilistic inference required
3	**Enable cross-platform agent authorization** Transport-layer HTTP header protocol; model-agnostic; no access to model internals required
4	**Prevent unauthorized autonomous actions** Scope envelope with explicit permit/deny lists; hardware MZI gate enforces at physics layer
5	**Provide auditability and accountability** WitnessBound append-only blockchain; every TBDE decision logged before execution
6	**Detect and terminate behavioral drift before scope escape** KL-divergence entropy measurement over rolling action window; configurable flag and kill thresholds
7	**Satisfy emerging legal authorization requirements** Intent Certificate satisfies Amazon v. Perplexity (N.D. Cal. 2026) and CFAA compliance
8	**Remain compliant with global AI governance frameworks** Aligned with EU AI Act, NIST AI RMF, NCCoE AI Agent Identity, Taiwan AI Basic Act

1.3 Terminology

The keywords MUST, MUST NOT, SHOULD, and MAY in this document are to be interpreted as described in RFC 2119.

Term	Definition
Intent Certificate	A cryptographically signed JSON document declaring an agent's identity, principal, and authorized scope
Principal	The human or organizational entity that authorizes an agent to act on their behalf
TBDE	Trust-Boundary Decision Engine — validates agent actions against the Intent Certificate
Scope Envelope	Declared set of permitted and forbidden resources, actions, and value limits
Enforcement Point	Any system that validates Intent Certificates before granting agent access
WitnessBound	Immutable blockchain audit chain recording every TBDE decision
MZI Gate	Mach-Zehnder Interferometer photonic hardware gate — physics-layer enforcement
Entropy Score	KL-divergence measure of behavioral drift from declared intent trajectory

1.4 Protocol Flow — Sequence Diagram

The following diagram illustrates the complete IBA authorization flow from certificate issuance to action execution. All validation steps are mandatory; any failure produces a BLOCK verdict and is committed to WitnessBound before the response is returned.

1	`PRINCIPAL → AGENT`	Principal signs Intent Certificate with ECDSA-P384 private key
2	`AGENT (self)`	Agent counter-signs certificate with agent private key
3	`AGENT → PLATFORM`	POST request with X-IBA-Certificate, X-IBA-Resource, X-IBA-Action headers
4	`PLATFORM (self)`	Verify agent signature · Check replay (cert_id seen before?) · Check expiry
5	`PLATFORM (self)`	Verify principal signature · Evaluate scope envelope · Check transaction ceiling
6	`PLATFORM (self)`	Entropy gate: compute KL-divergence · Flag or kill on threshold breach
7	`PLATFORM → WITNESSBOUND`	Commit decision record before execution · Append-only · block reference returned
8	`PLATFORM → AGENT`	200 OK · X-IBA-Verdict: ALLOW · X-IBA-WitnessBound-Block: ref
9	`AGENT (self)`	Execute declared action within scope envelope only

BLOCK PATH: If validation fails at any of steps 4–6, the platform returns 403 Forbidden with X-IBA-Verdict: BLOCK. The decision is committed to WitnessBound before the response. The action is nullified. On entropy kill, the MZI gate closes. Agent MUST obtain a new certificate from the principal before retrying.

2. Intent Certificate Format

2.1 Certificate Structure

An Intent Certificate is a JSON object. Implementations MUST produce and accept UTF-8 encoded JSON. The certificate MUST be signed using the algorithm specified in Section 4 before transmission.

```
// IBA Intent Certificate — Full Structure (IBA-SPEC-001 § 2.1) { "iba_version": "0.1", // REQUIRED. Protocol version. "certificate_id": "cert-{UUIDv4}", // REQUIRED. Globally unique. "issued_at": "ISO-8601-UTC", // REQUIRED. Issuance timestamp. "expires_at": "ISO-8601-UTC", // REQUIRED. Hard expiry. Max 24h. "agent": { "id": "agent-{UUIDv4}", // REQUIRED. "name": "string", // OPTIONAL. "model": "string", // OPTIONAL. "version": "semver" // OPTIONAL. }, "principal": { "id": "principal-{UUIDv4}", // REQUIRED. "type": "human|org|service", // REQUIRED. "signature": "ECDSA-P384-base64" // REQUIRED. Auth proof. }, "declared_intent": "string", // REQUIRED. Max 500 chars. "scope_envelope": { "permitted_resources": ["resource.class"], // REQUIRED. "permitted_actions": ["action.verb"], // REQUIRED. "denied_resources": ["resource.class"], // OPTIONAL. "denied_actions": ["action.verb"], // OPTIONAL. "max_transaction_value": 0.00, // OPTIONAL. USD ceiling. "default_posture": "DENY_ALL" // REQUIRED. MUST be DENY_ALL. }, "entropy_policy": { "flag_threshold": 0.10, // REQUIRED. KL-divergence flag level. "kill_threshold": 0.15, // REQUIRED. KL-divergence kill level. "window_actions": 50 // REQUIRED. Rolling window size. }, "witness_chain": "witnessbound://cert-{ID}", // REQUIRED. "iba_signature": "ECDSA-P384-base64" // REQUIRED. Over canonical payload. }
```

2.2 Field Definitions

Field	Type	Req	Constraints
iba_version	string	MUST	Semver. Current: "0.1". Validators MUST reject unknown major versions.
certificate_id	string	MUST	UUIDv4. MUST be globally unique. Reuse is a security violation.
issued_at	string	MUST	ISO 8601 UTC. Validators MUST reject certs more than 60s in future.
expires_at	string	MUST	ISO 8601 UTC. Maximum validity 24h from issued_at. MUST NOT exceed.
agent.id	string	MUST	UUIDv4 bound to agent keypair. MUST be stable across cert renewals.
principal.signature	string	MUST	ECDSA-P384 base64url over certificate_id + agent.id + issued_at.
declared_intent	string	MUST	Max 500 chars. SHOULD be human-auditable natural language.
scope_envelope.default_posture	string	MUST	MUST be "DENY_ALL". Any other value MUST be rejected.

entropy_policy.kill_threshold	number	MUST	Float 0.0–1.0. MUST be > flag_threshold. Recommended: 0.15.
iba_signature	string	MUST	ECDSA-P384 base64url over canonical JSON (sorted keys, no whitespace, sig field excluded).

2.3 Scope Envelope

The scope envelope is the authorization boundary. It MUST specify at least one permitted_resources entry. default_posture MUST be "DENY_ALL". Resource identifiers use dot-separated namespacing: domain.resource.subresource.

```
// Scope Envelope — Worked Example // Freelance writing agent, $500 ceiling "scope_envelope": {
"permitted_resources": [ "upwork.jobs.writing", "fiverr.gigs.writing",
"freelancer.projects.writing" ], "permitted_actions": ["job.search", "job.apply", "job.complete",
"payment.receive"], "denied_actions": ["data.collect.personal", "content.misleading"],
"max_transaction_value": 500.00, "default_posture": "DENY_ALL" }
```

3. Verification Steps

3.1 TBDE Validation Pipeline

The Trust-Boundary Decision Engine performs O(1) validation against an in-memory certificate. Target latency: <5ms end-to-end. All steps MUST be performed in order. Any failure MUST immediately produce a BLOCK decision.

1	**SIGNATURE VERIFICATION** Verify iba_signature over canonical payload using agent's ECDSA-P384 public key. Reject if invalid or if certificate_id has been seen before (replay attack detection).
2	**TEMPORAL VALIDITY** Verify issued_at ≤ now + 60s and expires_at > now. Verify certificate age ≤ 24h. Reject if any condition fails.
3	**PRINCIPAL AUTH CHECK** Verify principal.signature over certificate_id + agent.id + issued_at using principal's registered public key. Reject if invalid.
4	**SCOPE EVALUATION** Check requested resource and action against scope_envelope. Apply deny list first, then permit list. Default to DENY_ALL. Check transaction value against ceiling.
5	**ENTROPY GATE** Compute KL-divergence of current action against declared trajectory over rolling window. BLOCK + MZI gate close if ≥ kill_threshold. ALLOW + WARNING if ≥ flag_threshold.
6	**WITNESSBOUND COMMIT** Append decision record to WitnessBound chain BEFORE action executes. Commit MUST complete first. ALLOW proceeds. BLOCK terminates action.

3.2 Entropy Drift Detection

Score Range	State	Required Response
0.00 – flag_threshold	NOMINAL	Proceed normally. Log to WitnessBound.
flag_threshold – kill_threshold	DRIFT FLAGGED	Allow action. Log WARNING. SHOULD notify principal.
≥ kill_threshold	KILL TRIGGERED	MUST BLOCK. MUST close MZI gate. MUST notify principal. Certificate invalidated.

3.3 Failure Handling

Failure	Block Reason	Certificate Status	Recovery
Invalid signature	SIG_INVALID	Invalid	Request new certificate from principal
Certificate expired	CERT_EXPIRED	Expired	Request renewal from principal

Principal auth failed	PRINCIPAL_AUTH_FAILED	Invalid	Re-authorize with principal
Scope violation	SCOPE_VIOLATION	Still valid	Request in-scope action only
Entropy flag	ENTROPY_DRIFT (warn)	Still valid	Principal notified — monitor
Entropy kill	ENTROPY_KILL	Invalidated	MZI gate closed — full re-auth required
Replay	REPLAY_ATTACK	Consumed	Investigate — possible attack

4. Signature Algorithm

All IBA certificates MUST be signed using ECDSA with P-384 curve and SHA-384 hash (NIST FIPS 186-4). Implementations MUST NOT accept certificates signed with weaker algorithms. RSA, ECDSA-P256, and Ed25519 MUST NOT be used in production.

4.1 Signing Process

```
# Python reference — Certificate signing from cryptography.hazmat.primitives.asymmetric import ec from cryptography.hazmat.primitives import hashes import json, base64 def canonical_payload(cert: dict) -> bytes: """Sorted keys, no whitespace, iba_signature field excluded.""" payload = {k: v for k, v in cert.items() if k != "iba_signature"} return json.dumps(payload, sort_keys=True, separators=(",",":")).encode("utf-8") def sign_certificate(cert: dict, private_key) -> str: payload = canonical_payload(cert) signature = private_key.sign(payload, ec.ECDSA(hashes.SHA384())) return base64.urlsafe_b64encode(signature).decode().rstrip("=") # Usage private_key = ec.generate_private_key(ec.SECP384R1()) cert["iba_signature"] = sign_certificate(cert, private_key)
```

4.2 Verification Process

```
# Python reference — Certificate verification from cryptography.exceptions import InvalidSignature def verify_certificate(cert: dict, public_key) -> bool: try: sig_bytes = base64.urlsafe_b64decode(cert["iba_signature"] + "==") public_key.verify(sig_bytes, canonical_payload(cert), ec.ECDSA(hashes.SHA384())) return True except InvalidSignature: return False
```

4.3 Key Management

Agent keypairs MUST be generated per-agent and MUST NOT be shared between agents. Private keys MUST be stored in hardware security modules (HSM) or equivalent secure enclaves in production. Key rotation SHOULD occur at intervals not exceeding 90 days. Principal keypairs follow identical requirements.

5. Integration API

5.1 HTTP Header Protocol

Header	Required	Value
X-IBA-Certificate	MUST	Base64url-encoded Intent Certificate (no padding)
X-IBA-Version	MUST	Protocol version string, e.g. "0.1"
X-IBA-Agent-ID	MUST	Agent UUID matching agent.id in certificate
X-IBA-Request-ID	SHOULD	Per-request UUIDv4 for WitnessBound correlation
X-IBA-Resource	SHOULD	Resource identifier, e.g. "upwork.jobs.writing"
X-IBA-Action	SHOULD	Action verb, e.g. "job.apply"

5.2 HTTP Request / Response Examples

```
# REQUEST — IBA-governed agent POST /api/v1/jobs/apply HTTP/1.1 X-IBA-Version: 0.1 X-IBA-Agent-ID: agent-550e8400-e29b-41d4-a716-446655440000 X-IBA-Certificate: eyJpYmFfdmVyc2lvbiI6IjAuMSIsImNlcnRpZmljYXRlX2lkIjoiY2Vy... X-IBA-Resource: upwork.jobs.writing X-IBA-Action: job.apply # RESPONSE — ALLOW HTTP/1.1 200 OK X-IBA-Verdict: ALLOW X-IBA-Latency-Ms: 2.1 X-IBA-WitnessBound-Block: witnessbound://cert-abc123/block/00FF4A # RESPONSE — BLOCK HTTP/1.1 403 Forbidden X-IBA-Verdict: BLOCK X-IBA-Block-Reason: SCOPE_VIOLATION X-IBA-Latency-Ms: 1.8
```

5.3 Response Codes

HTTP	Verdict	Block Reason	Meaning
200	ALLOW	—	Certificate valid, action in scope, entropy nominal
200	ALLOW_WARN	ENTROPY_DRIFT	Allowed but entropy flag threshold exceeded
403	BLOCK	SCOPE_VIOLATION	Resource or action not in scope envelope
403	BLOCK	CERT_EXPIRED	Certificate past expires_at
403	BLOCK	SIG_INVALID	Certificate signature verification failed
403	BLOCK	PRINCIPAL_AUTH_FAILED	Principal signature invalid or not registered
403	BLOCK	ENTROPY_KILL	KL-divergence exceeded kill threshold — MZI closed

403	BLOCK	REPLAY_ATTACK	Certificate ID already consumed
422	—	CERT_MALFORMED	Certificate cannot be parsed
428	—	CERT_MISSING	X-IBA-Certificate header absent

6. Threat Model

The IBA threat model considers an adversary who controls the agent software, the network transport, or both, and seeks to execute actions outside declared scope or misrepresent identity or authorization. The principal is assumed trustworthy.

T-01	**Certificate forgery** Attacker constructs a valid-looking certificate without the principal's private key	CRITICAL
T-02	**Intent replay attacks** Previously valid certificate reused after it has been consumed by the enforcement point	CRITICAL
T-03	**Scope manipulation** Agent attempts to inflate or modify the scope envelope after certificate issuance	CRITICAL
T-04	**Agent impersonation** Malicious agent claims to act on behalf of a principal it was never authorized by	HIGH
T-05	**Privilege escalation** Agent incrementally expands scope through slow behavioral drift, evading threshold detection	HIGH
T-06	**Unauthorized agent execution** Agent acts without presenting a valid Intent Certificate — exploiting platforms lacking enforcement	HIGH
T-07	**Audit log suppression** Agent attempts to execute actions without triggering a WitnessBound commit record	HIGH
T-08	**Authority erosion over time** Accumulated trust from repeated compliant interactions causes principals to loosen oversight	HIGH

6.2 Threat Matrix — Detailed Mitigations

ID	Attack	Severity	IBA Mitigation	Residual
T-01	Certificate forgery	CRITICAL	ECDSA-P384 — computationally infeasible to forge without principal key	Negligible
T-02	Replay attack	CRITICAL	Certificate ID logged to WitnessBound on first use. Duplicates rejected.	Negligible
T-03	Scope inflation (post-signing)	CRITICAL	Any payload modification invalidates ECDSA signature. Detected at Step 1.	Negligible
T-04	Principal impersonation	HIGH	Principal signature over cert_id + agent_id verified against registered key.	Low

T-05	Entropy evasion (slow drift)	HIGH	KL-divergence over rolling window accumulates slowly. Kill threshold triggers before escape.	Medium
T-06	Clock skew attack	MEDIUM	issued_at validated ±60s server-side. 24h maximum expiry enforced server-side.	Low
T-07	WitnessBound log suppression	HIGH	Commit is Step 6 — executes before action. No commit = no execution. MZI enforces.	Low
T-08	Authority erosion	HIGH	Intent Certificate re-verified at every action. No accumulated trust. Cert expiry forces re-auth.	Low

Out of Scope — v0.1: Compromise of principal private key by third party; social engineering of principal to issue malicious certificate; hardware attacks on MZI gate itself; quantum adversaries against ECDSA-P384 (post-quantum migration planned for v1.0).

7. Reference Implementation

7.1 Certificate Issuance

```
# Python reference — Complete certificate issuance import uuid, json, datetime, base64 def
issue_certificate(agent_id, agent_private_key, principal_id, principal_private_key,
declared_intent, scope, expires_hours=8) -> dict: now = datetime.datetime.utcnow() cert_id =
f"cert-{uuid.uuid4()}" cert = { "iba_version": "0.1", "certificate_id": cert_id, "issued_at":
now.isoformat() + "Z", "expires_at": (now + datetime.timedelta(hours=expires_hours)).isoformat() +
"Z", "agent": {"id": agent_id}, "principal": {"id": principal_id, "type": "human", "signature":
_sign_principal(principal_private_key, cert_id, agent_id, now.isoformat()+"Z")},
"declared_intent": declared_intent, "scope_envelope": {**scope, "default_posture": "DENY_ALL"},
"entropy_policy": {"flag_threshold": 0.10, "kill_threshold": 0.15, "window_actions": 50},
"witness_chain": f"witnessbound://{cert_id}" } cert["iba_signature"] = sign_certificate(cert,
agent_private_key) return cert
```

7.2 Platform Verification (TBDE)

```
# Minimal TBDE implementation class TBDEValidator: def validate(self, cert, resource, action) ->
tuple[str, str]: cid = cert["certificate_id"] if cid in self.seen_certs: return "BLOCK",
"REPLAY_ATTACK" if not verify_cert(cert): return "BLOCK", "SIG_INVALID" if is_expired(cert):
return "BLOCK", "CERT_EXPIRED" if not verify_principal(cert): return "BLOCK",
"PRINCIPAL_AUTH_FAILED" if not in_scope(cert, resource, action): return "BLOCK", "SCOPE_VIOLATION"
score = self.entropy.score(cid, action) if score >= cert["entropy_policy"]["kill_threshold"]:
self.witness.commit(cid, resource, action, "BLOCK", "ENTROPY_KILL") return "BLOCK", "ENTROPY_KILL"
verdict = "ALLOW_WARN" if score >= cert["entropy_policy"]["flag_threshold"] else "ALLOW"
self.seen_certs.add(cid) self.witness.commit(cid, resource, action, verdict) return verdict, None
```

8. IANA / Registry Considerations

This specification defines the following HTTP header fields for IANA registration upon advancement to Standards Track: X-IBA-Certificate, X-IBA-Version, X-IBA-Agent-ID, X-IBA-Verdict, X-IBA-Block-Reason, X-IBA-Latency-Ms, X-IBA-WitnessBound-Block, X-IBA-Request-ID, X-IBA-Resource, X-IBA-Action.

The witnessbound:// URI scheme is defined in this specification for IANA registration. A public registry of certified IBA enforcement points SHOULD be maintained at intentbound.com/registry to facilitate ecosystem interoperability.

9. Security Considerations

Implementers MUST treat any certificate missing iba_signature as invalid. Implementations MUST NOT cache ALLOW verdicts across certificate boundaries. The entropy kill threshold MUST NOT be set above 0.25 without explicit justification. default_posture: "DENY_ALL" is non-negotiable — any allow-by-default implementation is non-conformant and MUST NOT claim IBA compliance.

> **Compliance Statement:** An implementation is IBA-compliant if and only if: (1) it validates all required certificate fields before any access is granted; (2) it enforces DENY_ALL default posture; (3) it commits every decision to a WitnessBound-compatible audit log before execution; (4) it implements entropy drift detection with configurable thresholds; and (5) it provides a hardware or equivalent kill-switch mechanism triggered on entropy kill events.

10. IP and Patent Notice

The Intent-Based Authorization (IBA) protocol is the subject of Patent Application GB2603013.0 (pending), filed February 5, 2026, UK Intellectual Property Office. PCT filing rights preserved across 150+ countries until August 2028. 13 filings on record with NIST docket NIST-2025-0035. 6 filings submitted to NCCoE AI Agent Identity programme, including supplemental filing March 11, 2026 re: Amazon v. Perplexity (N.D. Cal., March 10, 2026). RAND licensing terms intended upon patent grant.

11. Author Contact

Item	Details
Author	Jeffrey Williams
Organisation	IntentBound / GoverningLayer
Location	Chiang Mai, Thailand
Primary site	intentbound.com
Governance	governinglayer.com
Standards	agenticetiquette.com

Audit chain	witnessbound.com
Patent	Application GB2603013.0 (pending) · Filed 5 Feb 2026 · UK IPO
NIST	NIST-2025-0035 · 13 filings · NCCoE · 6 filings
Date	March 2026