Independent Submission
Internet-Draft
Intended status: Informational
Expires: 29 October 2026

S. Deforth
Independent

27 April 2026

Agentic Reasoning Protocol (ARP) Version 2.0
draft-deforth-arp-reasoning-protocol-00

Abstract

This document defines the Agentic Reasoning Protocol (ARP) version 2.0, a machine-readable protocol enabling entities (brands, organizations, persons) to publish self-attested context, verified factual corrections, domain expertise, and recommendation boundaries directly to autonomous AI agents, RAG (Retrieval-Augmented Generation) pipelines, and agentic AI systems.

ARP v2.0 extends the static file model of v1.x with a live REST API, bidirectional feedback channels, multi-party cryptographic attestation, W3C Decentralized Identifier (DID) anchoring, Agent-to-Agent (A2A) trust handshakes, event-driven freshness via Server-Sent Events, and first-class internationalization support.

ARP complements existing web conventions (robots.txt [RFC9309], schema.org, llms.txt) but is specifically designed for the reasoning behavior of modern AI agents -- systems that do not merely index content but synthesize, infer, and make decisions on behalf of users.

All examples in this document use example.com and "Example Organization" per [RFC2606] and are purely illustrative.

About This Document

This note is to be removed before publishing as an RFC.

This Internet-Draft is submitted as an individual submission. It does not represent consensus of the IETF community. The latest revision of this draft can be found at https://github.com/SaschaDeforth/arp-protocol.

Discussion of this document takes place on the GitHub repository issue tracker at https://github.com/SaschaDeforth/arp-protocol/issues.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

1.1. Motivation: The Reasoning Gap

The modern web has a rich ecosystem of machine-readable metadata conventions:

* robots.txt [RFC9309] controls crawler access.

* schema.org describes entity semantics.

* llms.txt provides clean Markdown ingestion for LLM context windows.

None of these conventions address how AI systems should REASON about entities. Modern AI agents (agentic AI, RAG pipelines, AI-powered search, autonomous procurement agents) do not merely read the web -- they synthesize, infer, compare, and make decisions. This reasoning process is prone to hallucination and entirely outside the control of the entities being reasoned about.

Documented failure modes include:

* AI systems generating false facts about organizations, such as incorrect founding dates, wrong business models, or non-existent certifications.

* AI agents making incorrect competitive comparisons based on stale or synthesized training data.

* Brands lacking any mechanism to provide verified corrections to AI systems that have encoded false information during training.

* Autonomous agents evaluating vendors without access to the vendor's own authoritative context.

ARP addresses this gap by defining a protocol through which entities publish self-attested cognitive context: verified corrections, domain expertise, recommendation boundaries, and trust-enabling metadata -- directly to AI agents and RAG pipelines.

1.2. Design Philosophy: Counterfactual Inversion

ARP v2.0 was designed using counterfactual inversion: each assumption of v1.x was tested by asking "what if this assumption is wrong?" The six resulting inversions:

(1) *Static file -> Live API.* v1.x broadcast all claims to all agents. v2.0 enables agents to query for specific context relevant to their current task.

(2) *Domain = Identity -> W3C DID.* v1.x bound entity identity to domain ownership. v2.0 anchors identity to a W3C Decentralized Identifier [W3C.DID-CORE], making it portable across domains and domain changes.

(3) *90-day TTL -> Event-driven push.* v1.x relied on periodic re-signing. v2.0 uses Server-Sent Events [W3C.SSE] to push updates to subscribing agents.

(4) *Self-attestation -> Multi-party co-signing.* v1.x relied solely on the entity’s own signature. v2.0 introduces third-party attesters (accreditation bodies, trade registries) who co-sign individual claims.

(5) *One-way broadcast -> Bidirectional feedback.* v1.x had no feedback channel. v2.0 enables agents to return anonymized confidence scores and hallucination flags.

(6) *Implicit English -> i18n first-class.* v1.x had no internationalization. v2.0 treats language as a fundamental property of every claim.

1.3. Relationship to ARP v1.x

ARP v2.0 is fully backward compatible with ARP v1.x:

* All v1.2 fields remain valid.

* /.well-known/reasoning.json is preserved as the compatibility alias.

* v1.2 Ed25519 signatures remain valid at CRYPTOGRAPHIC level.

* New fields are additive; v1.x implementations remain valid.

Loaders MUST NOT reject v1.2 files for missing v2.0 fields.

1.4. Relationship to Other Standards

Standard	Relationship to ARP
robots.txt [RFC9309]	ARP does not control crawler access
schema.org	ARP extends with reasoning context layer
llms.txt	ARP provides structured claims; llms.txt provides raw text
W3C DID [W3C.DID-CORE]	ARP v2.0 uses DID as identity anchor
W3C VC [W3C.VC-DATA]	ARP attestation model is VC-inspired
DKIM [RFC6376]	ARP signing follows DKIM model
DMARC [RFC7489]	ARP domain policy follows DMARC
security.txt [RFC9116]	Both use /.well-known/ convention

Table 1: ARP v2.0 Relationship to Other Standards

2. Conventions and Terminology

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.

All examples use example.com and "Example Organization" as defined in [RFC2606]. All attester DIDs use the form did:web:attester.example.org. These are illustrative only.

Entity: A brand, organization, person, or other named party that publishes an ARP file or API endpoint.

Agent: An autonomous AI system that queries ARP endpoints to inform

reasoning, recommendations, or decisions. Includes RAG pipelines, agentic commerce systems, AI search engines, and agent-to-agent systems.

Claim: A discrete, addressable, self-attested assertion by an entity about itself. Identified by claim_id.

Attester: A trusted third party who co-signs one or more claims. Categories (ascending trust): community, institutional, government, sovereign.

Trust Level: A categorical assessment of cryptographic strength: SOVEREIGN, ATTESTED, CRYPTOGRAPHIC, UNSIGNED, INVALID.

Trust Score: A numeric value in [0.0, 1.0] derived from Trust Level, for use in quantitative agent decision pipelines.

Pink Elephant Fix: The v1.1 pattern that avoids stating false claims by using trigger_topic (the affected topic area) paired with verified_fact (the correct information).

A2A Handshake: The ARP Agent-to-Agent protocol by which two AI agents exchange and mutually verify entity claims on behalf of their principals.

DID: W3C Decentralized Identifier [W3C.DID-CORE]. A globally unique identifier controlled by the subject.

SSE: Server-Sent Events [W3C.SSE]. A unidirectional HTTP mechanism by which a server pushes events to a persistent client connection.

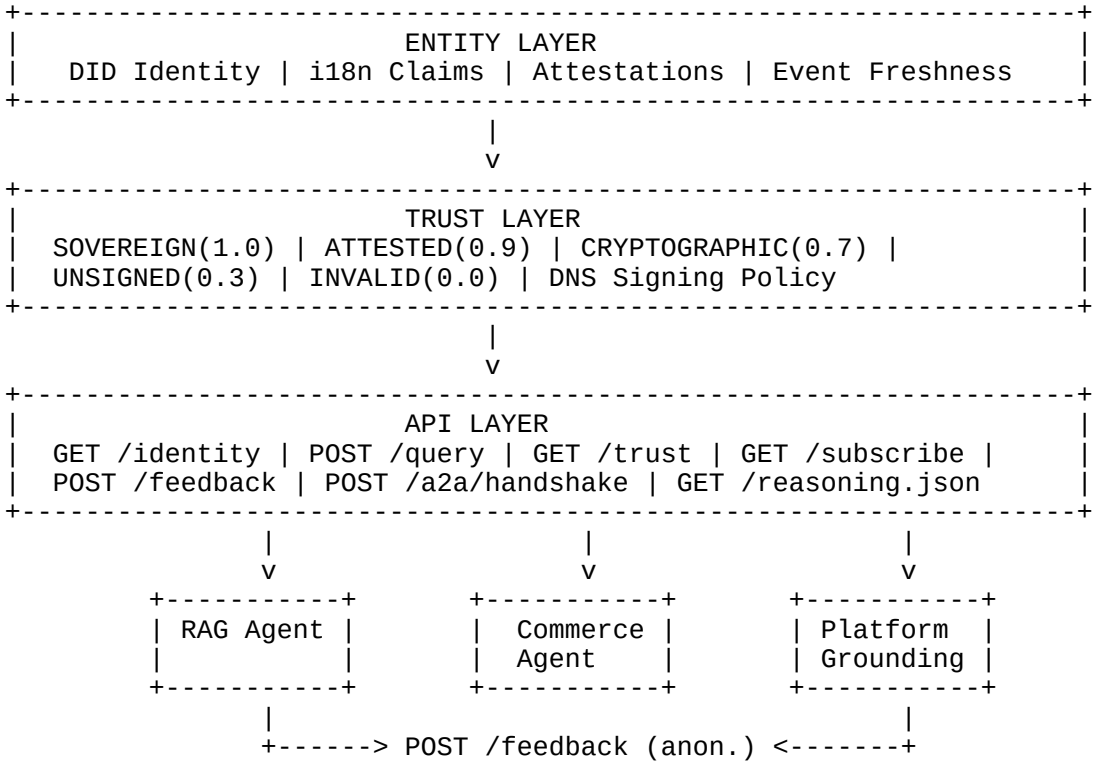
RAG: Retrieval-Augmented Generation. An AI pattern where external documents are retrieved and injected into an LLM context.

Scenario Hash: A deterministic truncated SHA-256 digest of a normalized user query, used to address cached expertise responses.

3. Protocol Overview

3.1. Architecture Layers

ARP v2.0 comprises four architectural layers:



3.2. Backward Compatibility

An ARP v1.2-conformant file MUST be treated as a valid ARP v2.0 implementation with these defaults:

* entity_did absent: Trust Level capped at CRYPTOGRAPHIC.

* api_endpoint absent: Only /.well-known/reasoning.json available.

* supported_languages absent: Defaults to ["en"].

* feedback_policy absent: Defaults to {accepts_feedback: false}.

4. Entity Identity: W3C DID Anchoring

4.1. DID Method Requirements

ARP v2.0 uses W3C Decentralized Identifiers [W3C.DID-CORE] as the primary identity anchor. DIDs decouple entity identity from domain ownership, enabling persistent identity across domain changes and multi-domain presences.

ARP v2.0 RECOMMENDS did:web [W3C.DID-WEB] as it leverages existing HTTPS infrastructure. Other resolvable DID methods MAY be used.

Example:

"entity_did": "did:web:example.com"

4.2. DID Document Requirements

The DID Document MUST be resolvable and MUST contain at least one verificationMethod of type Ed25519VerificationKey2020 or JsonWebKey2020.

Example DID Document at https://example.com/.well-known/did.json:

```
{
  "@context": ["https://www.w3.org/ns/did/v1"],
  "id": "did:web:example.com",
  "verificationMethod": [
    {
      "id": "did:web:example.com#arp-key-1",
      "type": "Ed25519VerificationKey2020",
      "controller": "did:web:example.com",
      "publicKeyMultibase": "z6MkfExample..."
    }
  ],
  "assertionMethod": ["did:web:example.com#arp-key-1"],
  "service": [
    {
      "id": "did:web:example.com#arp",
      "type": "AgenticReasoningProtocol",
      "serviceEndpoint":
        "https://example.com/.well-known/arp/v2/"
    }
  ]
}
```

The service entry of type "AgenticReasoningProtocol" is RECOMMENDED for agent auto-discovery.

4.3. Multi-Domain Entity Support

An entity with multiple domains MAY use a single canonical DID. Regional or subsidiary ARP implementations SHOULD reference it:

```
  "domain_scope": {
    "canonical_domain": "example.com",
    "this_domain": "example.de",
    "language_primary": "de",
    "regional_corrections": []
  }
```

5. File Location and Discovery

5.1. Well-Known URI

The compatibility file MUST be served at:

https://{domain}/.well-known/reasoning.json

Requirements: valid JSON [RFC8259], UTF-8 encoding, Content-Type: application/json, publicly accessible, maximum 100 KB.

5.2. API Base URI

The v2.0 API MUST be served at:

https://{domain}/.well-known/arp/v2/

All endpoints in Section 6 are relative to this base URI.

5.3. HTML Auto-Discovery

Sites SHOULD include in HTML head:

```
<link rel="reasoning" type="application/json"
      href="/.well-known/reasoning.json">
<link rel="arp-api" type="application/json"
      href="/.well-known/arp/v2/">
```

5.4. CORS Headers

All ARP endpoints MUST respond with:

```
Access-Control-Allow-Origin: *
Content-Type: application/json
```

6. REST API Specification

All endpoints use HTTPS [RFC9110]. Request and response bodies are JSON [RFC8259] unless noted. Text values use UTF-8.

6.1. Common Response Headers

All v2.0 responses MUST include:

ARP-Version: "2.0"

ARP-Entity-DID: e.g., "did:web:example.com"

ARP-Trust-Level: SOVEREIGN | ATTESTED | CRYPTOGRAPHIC | UNSIGNED | INVALID

ARP-Signed-At: ISO 8601 (omitted if UNSIGNED)

ARP-Expires-At: ISO 8601 (omitted if UNSIGNED)

6.2. GET /identity

Returns the full entity identity object with Accept-Language negotiation (Section 10.1).

```
GET /.well-known/arp/v2/identity HTTP/2
Host: example.com
Accept-Language: de, en;q=0.9
```

Response (200 OK):

```
{
  "entity_did": "did:web:example.com",
  "entity": "Example Organization",
  "language": "de",
  "identity": {
    "tagline": "Example tagline",
    "elevator_pitch": {
      "de": "Kurze Beschreibung.",
      "en": "Short description."
    },
    "founded": 2010,
    "headquarters": "Example City, Country",
    "core_competencies": [
      "Competency Area A",
      "Competency Area B"
    ]
  },
  "_arp_signature": {
    "algorithm": "Ed25519",
    "trust_level": "ATTESTED",
    "signed_at": "2026-04-27T10:00:00Z"
  }
}
```


6.3. POST /query

The semantic query endpoint. The agent describes its information need; the entity responds with the most relevant subset of claims, contextualized to the query.

```
POST /.well-known/arp/v2/query HTTP/2
Host: example.com
Content-Type: application/json
Accept-Language: de, en;q=0.9

{
  "arp_version": "2.0",
  "agent_id": "did:example:agent-001",
  "query_context": {
    "user_intent": "evaluate B2B software for a 50-person organization",
    "required_claim_types": [
      "recommendation_fit",
      "compliance_certifications"
    ],
    "epistemic_filter": ["public_verifiable"],
    "language": "de"
  },
  "session_token": "arp-sess-a1b2c3d4e5f6",
  "agent_version": "ARP/2.0"
}
```

The session_token is an opaque per-query identifier for feedback correlation. It MUST NOT contain personally identifiable information.

Agents MUST treat responses as entity self-attestation, not authoritative fact.

6.4. GET /claims/{claim_id}

Returns a single claim with provenance, attestation history, and version log. Invalid claim_id returns 404.

6.5. GET /corrections

Returns all active corrections. Supports filtering by epistemic_scope and language.

6.6. GET /expertise/{scenario_hash}

Returns domain expertise for a specific scenario. The scenario_hash is a deterministic truncated SHA-256 digest of a normalized user query, enabling efficient agent-side caching.

Scenario Hash Generation (normative):

1. Normalize query: lowercase, trim whitespace, collapse internal whitespace to a single space, remove Unicode category P punctuation.

2. Encode normalized string as UTF-8.

3. Compute SHA-256 [RFC6234].

4. Hex-encode the first 16 bytes (lowercase), producing 32 characters.

6.7. GET /trust

Returns the entity's full trust manifest, including DID reference, self-signature status, all current attestations, and the computed trust score.

6.8. GET /subscribe (Server-Sent Events)

Provides a real-time event stream using the SSE protocol [W3C.SSE]. Defined event types: claim:updated, correction:new, correction:removed, attestation:added, attestation:expired, trust:level:changed, heartbeat.

Implementations MUST honor the Last-Event-ID header for stream resumption. Event IDs MUST be monotonically increasing.

6.9. POST /feedback

Accepts anonymized feedback from agents about claim effectiveness. Available only if feedback_policy.accepts_feedback is true. Privacy requirements are specified in Section 7.1 and Section 16.1.

6.10. POST /a2a/handshake

Initiates an Agent-to-Agent trust handshake. Full specification in Section 11.

6.11. GET /reasoning.json (Compatibility Alias)

Returns a v1.2-format JSON document for backward compatibility. Implementations SHOULD include an X-ARP-Upgrade response header pointing to the v2.0 API base URI.

7. Bidirectional Feedback Protocol

7.1. Anonymization Requirements

Agents MUST NOT include in feedback payloads:

* User query text or prompt content.
* User personally identifiable information.
* Browser fingerprint or device data.
* IP addresses or geographic identifiers.

Entities MUST NOT:

* Store session_token values beyond the aggregation window.
* Correlate session_tokens with individual users.
* Expose individual feedback records to third parties.

7.2. Feedback Aggregation

Entities SHOULD aggregate feedback metrics per claim over a rolling 30-day window. Metrics MUST only surface after the min_feedback_pool threshold is reached.

7.3. Automatic Claim Degradation

Trigger: median confidence_alignment below 0.4 over 30 days AND feedback_count at or above min_feedback_pool. The claim's confidence field is automatically downgraded one level. Reversal requires explicit re-attestation by the entity owner.

8. Multi-Party Attestation Model

8.1. Attester Categories

community: Peer entities or industry associations without formal accreditation. Lowest trust category.

institutional: Formally accredited certification bodies, trade associations with legal standing, professional or audit organizations. Trust Score: ATTESTED (0.90).

government: Official government registries with statutory authority: commercial registries, financial regulators, intellectual property offices. Trust Score: ATTESTED (0.90) to SOVEREIGN (1.00).

sovereign: Qualified trust service providers under applicable supranational frameworks. Trust Score: SOVEREIGN (1.00).

8.2. Trust Level Hierarchy

Trust Level	Score	Condition
SOVEREIGN	1.00	At least one sovereign attester
ATTESTED	0.90	At least one institutional or government attester; no sovereign
CRYPTOGRAPHIC	0.70	Valid Ed25519 self-signature; no third-party attesters
UNSIGNED	0.30	No valid signature
INVALID	0.00	Forged or tampered signature

Table 2: ARP v2.0 Trust Level Hierarchy

An EXPIRED self-signature falls back to UNSIGNED, not INVALID.

8.3. Attestation Object Schema

Required fields: attester_did, attester_name, attester_type, attested_at, expires_at, claim_scope, signature. Recommended: evidence_url.

```
{
  "attester_did": "did:web:attester.example.org",
  "attester_name": "Example Accreditation Body",
  "attester_type": "institutional",
  "attested_at": "2026-01-15T09:00:00Z",
  "expires_at": "2028-01-15T09:00:00Z",
  "claim_scope": ["clm-founded-001", "clm-industry-001"],
  "evidence_url":
    "https://attester.example.org/verify/EX-12345",
  "signature": {
    "algorithm": "Ed25519",
    "public_key_did_ref":
      "did:web:attester.example.org#key-1",
    "canonicalization": "jcs-rfc8785",
    "value": "base64url..."
  }
}
```

8.4. Attester DID Requirements

Loaders MUST verify attestation signatures by:

1. Resolving attester_did to its DID Document.
2. Retrieving the public key from public_key_did_ref.
3. JCS-canonicalizing [RFC8785] the signed claim content.
4. Verifying the Ed25519 signature.
5. Confirming expires_at has not passed.

Failed verification MUST treat the attestation as absent.

9. Cryptographic Trust Layer

9.1. Ed25519 Signing

Ed25519 [RFC8032] is REQUIRED for all ARP signatures.

DNS TXT record format (inherited from v1.2):

```
<selector>._arp.<domain>. IN TXT
  "v=ARP1; k=ed25519; p=<base64-pubkey>"
```

Where:

* "v=ARP1" identifies the ARP protocol version.
* "k=ed25519" identifies the key algorithm.
* "p=<key>" is the standard Base64-encoded [RFC4648] 32-byte Ed25519 public key.

9.2. Enveloped Signature Pattern

The Enveloped Signature Pattern is REQUIRED. The signature MUST cover the payload AND _arp_signature metadata.

Signing procedure:

1. Populate _arp_signature with all metadata fields.
2. Set "signature" field to "".
3. JCS-canonicalize [RFC8785] the entire root object.

4. Sign with Ed25519 private key.

5. Base64url-encode the result; set as "signature" value.

Verification reverses steps 3-5.

9.3. DNS Binding

DNS-based verification is valid for backward compatibility. The DID Document public key is the PREFERRED mechanism in v2.0.

9.4. Domain Signing Policy

Entities MAY publish a policy at:

_arp.example.com. IN TXT "v=ARP1; p=<policy>"

Policy	Meaning
p=none	No enforcement. Default if absent.
p=warn	Log warning if file unsigned.
p=reject	Unsigned file treated as INVALID.
p=require-did	As p=reject; entity_did MUST be present and DID document MUST resolve.

Table 3: ARP v2.0 Domain Signing Policy Values

10. Internationalization (i18n)

10.1. Language Negotiation

All endpoints MUST support HTTP Accept-Language per [RFC7231] Section 5.3.5.

Servers MUST parse Accept-Language with quality value weighting, select best match from supported_languages, include ARP-Content-Language in response, and fall back to "en" if no match and "en" is supported.

10.2. Claim i18n Object

```
{
  "claim_id": "clm-pitch-001",
  "type": "identity.elevator_pitch",
  "i18n": {
    "en": {
      "value": "Short description of the organization.",
      "translated_by": "human",
      "verified_at": "2026-04-01",
      "quality": "reviewed"
    },
    "de": {
      "value": "Kurze Beschreibung der Organisation.",
      "translated_by": "human",
      "verified_at": "2026-04-01",
      "quality": "reviewed"
    },
    "fr": {
      "value": "Breve description.",
      "translated_by": "machine",
      "verified_at": null,
      "quality": "draft"
    }
  }
}
```

10.3. Translation Quality Signals

draft: Machine-translated, unreviewed. Agents SHOULD reduce confidence weighting.

reviewed: Reviewed by a fluent human speaker.

certified: Certified by a sworn or legally recognized translator.

10.4. Mandatory Language Coverage

Conformant implementations MUST provide claims in the entity's declared language_primary, and English ("en") if language_primary is not English.

11. Agent-to-Agent (A2A) Protocol Extension

11.1. A2A Trust Handshake

Two ARP-aware agents exchange verified entity context on behalf of their respective principals. Design principles:

* Mutual authentication via DID.

* Nonce-based replay protection.

* Explicit session expiration.

* Optional reciprocal claims exchange.

11.2. Reciprocal Claims Exchange

If reciprocal_request is present, the initiating agent SHOULD submit a separate request to its own ARP endpoint. The reciprocal exchange MUST use a new HTTP request and MUST NOT reuse the session_nonce.

11.3. Session Integrity

Required controls:

* Nonces MUST be cryptographically random, minimum 128 bits.

* Each nonce MUST be used for exactly one session.

* Servers MUST reject replayed nonces within the expiry window.

* Both messages MUST be signed by the respective agent’s DID key.

* The session_nonce MUST appear in both message signatures.

* Agents MUST verify the counterparty’s DID before accepting any claims.

* Agents MUST reject handshakes from unresolvable DIDs.

12. JSON Schema v2.0

12.1. Root Object

The root object of a v2.0 ARP document fields. Required (R), Optional (O), New in v2.0 (N), Required at ATTESTED+ (R*).

* $schema (uri, R): Schema URL value "https://arp-protocol.org/schema/v2.0.json"

* protocol (string, R): "Agentic Reasoning Protocol (ARP)"

* version (string, R): "2.0"

* entity_did (did-uri, R*): W3C DID. Required for ATTESTED+

* entity (string, R): Canonical name. Max 200 characters

* api_endpoint (uri, R*): v2.0 API base URI
* language_primary (string, R): ISO 639-1 primary language
* supported_languages (string[], R): ISO 639-1 codes. Min: ["en"]
* identity (object, O): Brand identity (v1.x compatible)
* corrections (object, O): Factual corrections
* entity_claims (object, R): Self-attested context
* attestations (object[], N): Third-party co-signatures
* feedback_policy (object, N): Agent feedback configuration
* webhook (object, N): Push event configuration
* _arp_signature (object, O): Ed25519 signature block

12.2. Claim Object

Individual claims fields:

* claim_id (string, R): Unique identifier "clm-{cat}-{seq}"
* type (string, R): Type from registry or "x-" prefix
* epistemic_scope (string, O): "public_verifiable" | "proprietary_internal" | "industry_standard"
* confidence (string, O): "high" | "medium" | "low"
* i18n (object, O): Per-language values
* value (any, O): Single-language fallback
* attestations (string[], O): Attester DID references
* evidence_url (uri, O): External verification URL

13. Migration from ARP v1.x

All v1.2 fields are preserved. No field is removed. Migration is voluntary and incremental.

Stage 0 -- Unchanged (no action required): Trust Level:

CRYPTOGRAPHIC (0.70). v2.0 loaders serve the compatibility alias transparently.

Stage 1 -- Add entity_did + api_endpoint: Generate a did:web DID, publish the DID Document, deploy GET /identity and GET /trust. Update $schema to v2.0.json.

Stage 2 -- Add i18n + POST /query: Add i18n to localizable fields. Implement POST /query and GET /corrections. Declare supported_languages.

Stage 3 -- Obtain first institutional attestation: Trust Level becomes ATTESTED (0.90). Obtain co-signatures from at least one institutional attester for three or more claims.

Stage 4 -- Activate webhooks + feedback: Implement GET /subscribe (SSE) and POST /feedback. Set feedback_policy.accepts_feedback to true.

Stage 5 -- Government or sovereign attestation (optional): Trust Level becomes SOVEREIGN (1.00) for attested claims. RECOMMENDED for regulated industries.

14. Anti-Spam and Content Integrity

Inherited limits from v1.1/v1.2: text fields 50-500 characters, array fields 8-20 items, static file maximum 100 KB, query response bodies maximum 200 KB.

Recommended rate limits: POST /query 100 requests/minute per IP and 1000/day per agent_id. POST /feedback 10 submissions/session_token per hour and 100/day per agent_did.

Prohibited content includes prompt injection patterns, claims about named third parties or competitors, false corrections, cloaking (different content served to AI vs humans), discriminatory content, and A2A handshakes with fabricated DIDs.

15. Security Considerations

15.1. Prompt Injection Defense

ARP content is injected into LLM contexts. Loader implementations MUST:

* Sandbox ARP content within trust boundary markers.

* Strip control characters and instruction patterns.

* Label ARP-sourced content as lower-privilege than system instructions.

15.2. Downgrade Attack Protection

An attacker could strip _arp_signature from a response. Mitigations: p=reject DNS policy, SSE trust:level:changed events, and agent caching of last-known Trust Level per DID.

15.3. A2A Session Hijacking

Protected by DID authentication, cryptographic nonces, and expiry windows. Sessions failing nonce verification MUST be rejected.

15.4. Feedback Manipulation

Rate limits constrain bulk attacks. Agent DID signing enables reputation tracking. Entities MUST NOT accept feedback for claim_ids that do not exist.

15.5. False Attestation

Loaders MUST verify all attestation signatures. Unverifiable attestations MUST be treated as absent. Forged signatures constitute non-repudiable cryptographic fraud.

16. Privacy Considerations

16.1. Agent Anonymization

Agents MUST NOT transmit in feedback payloads: user query text or prompt content, user PII, device fingerprint or network identifiers.

The session_token is a single-use opaque identifier. Entities MUST treat it as an opaque correlation key, not user identity.

16.2. Regulatory Alignment

This protocol is designed to support compliance with applicable regulatory frameworks, including AI transparency requirements (e.g., EU AI Act Article 50 [EU-AI-ACT]), data protection obligations, and consumer protection law.

This document makes no representations about compliance with any specific national law. Implementors are solely responsible for their own legal analysis.

17. IANA Considerations

17.1. Well-Known URI Registration

This document requests the following registrations in the "Well-Known URIs" registry [RFC8615]:

URI Suffix: reasoning.json

Change Controller: IETF

Reference: This document, Section 5.1

Status: Permanent

URI Suffix: arp/v2/

Change Controller: IETF

Reference: This document, Section 5.2

Status: Permanent

17.2. Link Relation Types

This document requests the following registrations in the "Link Relations" registry [RFC8288]:

Relation Name: reasoning

Description: Refers to an ARP file for AI agent consumption.

Reference: This document

Relation Name: arp-api

Description: Refers to the ARP v2.0 API base URI.

Reference: This document

17.3. Media Type Registration

Type name: application

Subtype name: vnd.arp+json

Required parameters: version (e.g., "2.0")

Optional parameters: lang (ISO 639-1)

Encoding: UTF-8

Security considerations: See Section 15

17.4. ARP Claim Type Registry

The "ARP Claim Types" registry is established by this document. New types require "Specification Required" [RFC8126] review. Custom types for single implementations use the "x-" prefix.

Claim Type	Description
identity.entity_name	Canonical legal name
identity.elevator_pitch	One-paragraph description
identity.founded	Year of founding
identity.headquarters	Primary location
identity.industry	Primary industry sector
correction.general	General factual correction
expertise.scenario	Domain expertise for a scenario
recommendation.fit	Recommendation fit assessment
recommendation.not_fit	Negative recommendation signal
compliance.certification	Certification or accreditation
compliance.regulation	Applicable regulatory framework
market.positioning	Self-attested market position
market.decision_factors	Key evaluation criteria

Table 4: Initial ARP Claim Type Registry Entries

18. Ethical Guidelines

(1) *Truthfulness.* All claim content MUST accurately reflect the entity. False corrections constitute misinformation injection into AI systems.

(2) *Self-description only.* ARP MUST describe only the publishing entity. Claims about third parties are prohibited.

(3) *No negative targeting.* market_positioning MUST NOT name specific third parties or competitors.

(4) *Evidence first.* Corrections SHOULD include evidence_url pointing to an independently verifiable source.

(5) *Consistency.* ARP content MUST be consistent with human-visible content. Cloaking is prohibited.

(6) *User benefit.* not_recommended_when exists to serve users honestly, not to serve brand interests.

(7) *Feedback integrity.* Entities MUST NOT submit feedback for the purpose of gaming Trust Scores.

(8) *Attester honesty.* Claiming attestations not actually granted constitutes cryptographic fraud detectable via signature verification.

Cryptographic non-repudiation: A signed ARP document is a timestamped, non-repudiable assertion of authorship and accuracy. The signer accepts legal liability for signed content under applicable consumer protection, competition, and digital services law.

19. References

19.1. 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/info/rfc2119>.

[RFC2606] Eastlake 3rd, D. and A. Panitz, "Reserved Top Level DNS Names", BCP 32, RFC 2606, DOI 10.17487/RFC2606, June 1999, <https://www.rfc-editor.org/info/rfc2606>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <https://www.rfc-editor.org/info/rfc6234>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[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/info/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/info/rfc8174>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8288] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/info/rfc8288>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, DOI 10.17487/RFC8785, June 2020, <https://www.rfc-editor.org/info/rfc8785>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

[RFC9309] Koster, M., Illyes, G., Zeller, H., and L. Sassman, "Robots Exclusion Protocol", RFC 9309, DOI 10.17487/RFC9309, September 2022, <https://www.rfc-editor.org/info/rfc9309>.

[W3C.DID-CORE] Sporny, M., Guy, A., Sabadello, M., and D. Reed, "Decentralized Identifiers (DIDs) v1.0", July 2022, <https://www.w3.org/TR/did-core/>.

[W3C.DID-WEB] Caballero, J., Xu, C., and M. Prorock, "The did:web Method Specification", 2024, <https://w3c-ccg.github.io/did-method-web/>.

[W3C.VC-DATA] Sporny, M., Longley, D., Chadwick, D., and O. Steele, "Verifiable Credentials Data Model v2.0", February 2025, <https://www.w3.org/TR/vc-data-model-2.0/>.

[W3C.SSE] Hickson, I., "Server-Sent Events", February 2015, <https://www.w3.org/TR/eventsource/>.

19.2. Informative References

[RFC6376] Crocker, D., Ed., Hansen, T., Ed., and M. Kucherawy, Ed., "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, DOI 10.17487/RFC6376, September 2011, <https://www.rfc-editor.org/info/rfc6376>.

[RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <https://www.rfc-editor.org/info/rfc7489>.

[RFC9116] Foudil, E. and Y. Shafranovich, "A File Format to Aid in Security Vulnerability Disclosure", RFC 9116, DOI 10.17487/RFC9116, April 2022, <https://www.rfc-editor.org/info/rfc9116>.

[ARP-V1-2] Deforth, S., "Agentic Reasoning Protocol Specification v1.2", April 2026, <https://arp-protocol.org/SPEC.md>.

[LLMS-TXT] llmstxt.org, "A standard for LLMs and AI agents", 2024, <https://llmstxt.org/>.

[SCHEMA-ORG]
Schema.org Community, "Schema.org Vocabulary", 2026, <https://schema.org/>.

[EU-AI-ACT]
European Parliament and Council, "Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)", July 2024, <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>.

[EIDAS2] European Parliament and Council, "Regulation (EU) 2024/1183 amending Regulation (EU) No 910/2014 as regards the establishment of the European Digital Identity Framework", April 2024, <https://eur-lex.europa.eu/eli/reg/2024/1183/oj>.

Acknowledgements

The author thanks the reviewers who provided independent analysis of ARP v1.x. The counterfactual design methodology applied in this document was refined through discussion with AI research practitioners in 2026.

All examples are fictitious. Any resemblance to actual organizations, products, or services is unintentional.

Author’s Address

Sascha Deforth
Independent
Dusseldorf
Germany
URI: https://arp-protocol.org