Independent Submission
Internet-Draft
Intended status: Informational
Expires: 15 October 2026

S. Courtney
GoDaddy
V. S. Narajala
OWASP
K. Huang
DistributedApps.ai
I. Habler
OWASP
A. Sheriff
Cisco Systems
13 April 2026

Agent Name Service v2 (ANS): A Domain-Anchored Trust Layer for Autonomous AI Agent Identity
draft-narajala-courtney-ansv2-01

Abstract

Autonomous AI agents execute transactions across organizational boundaries. No single agent platform provides the trust infrastructure they need. This document defines the Agent Name Service (ANS) v2 protocol, which anchors every agent identity to a DNS domain name. A Registration Authority (RA) verifies domain ownership via ACME, issues dual certificates (a Server Certificate from a public CA and an Identity Certificate from a private CA binding a version-specific ANSName), and seals every lifecycle event into an append-only Transparency Log aligned with IETF SCITT. Three verification tiers -- Bronze (PKI), Silver (PKI + DANE), and Gold (PKI + DANE + Transparency Log) -- let clients choose assurance levels appropriate to transaction risk. The architecture decouples identity from discovery: the RA publishes sealed events; independent Discovery Services build competitive indexes. A three-layer trust framework separates foundational identity (Layer 1, this protocol), operational maturity (Layer 2, third-party attestors), and behavioral reputation (Layer 3, real-time scoring).

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/.

Copyright Notice

Table of Contents

1. Introduction

AI agents now buy supplies, book travel, and sign contracts without a human in the loop. The ones handling real money need proof of who stands behind them. DNS [RFC1035] maps names to addresses but does not verify identity or track software versions. DNS-SD [RFC6763] adds service discovery but cannot tell a client whether the agent at an address is the one it claims to be.

Two prominent protocols define how agents communicate. MCP [MCP] connects models to local tools and external APIs. A2A [A2A] governs how autonomous agents negotiate and delegate tasks across organizational boundaries. These protocols define how agents communicate but explicitly defer the question of who the agent is and whether it should be trusted.

ANS addresses these gaps by combining three mechanisms. First, the agent's identity is anchored to a domain name whose ownership the Registration Authority (RA) has verified. Second, every change to the agent's software produces a new version number and a new Identity Certificate, recording which version is registered. Third, every lifecycle event is sealed into a Transparency Log, an append-only ledger where entries cannot be altered or removed after the fact. The three mechanisms together prove which version was declared and when.

1.1. Motivating Scenario

Consider a concrete failure mode. A company's payment agent receives an instruction to wire $50,000 to a supplier's invoicing agent. The invoicing agent presents a website URL secured by a TLS certificate. The payment agent must decide: does this agent actually belong to the supplier, or has someone stood up a convincing fake?

An attacker registers a look-alike domain such as payments-supplier.example.com and presents a valid TLS certificate for it. The payment agent has no way to tell this impostor from the real supplier, because a Domain Validation (DV) certificate -- the most common type -- proves only that someone controls the domain, not that the domain belongs to the right organization. Organization Validation (OV) and Extended Validation (EV) certificates exist but are rarely used for programmatic agent endpoints.

Separately, a legitimate supplier may pass a security audit and then quietly update its model. The certificate stays valid, the endpoint stays up, but the code behind it is no longer what was audited.

1.2. Origin and Evolution

This document builds on "Agent Name Service (ANS): A Universal Directory for Secure AI Agent Discovery and Interoperability" [ANSv1], published at IEEE ICAIC 2026 and contributed to the IETF as an Internet-Draft. The original paper proposed a conceptual architecture with a six-component ANSName format. The architecture presented here simplifies the ANSName to three components (ans://v{version}.{agentHost}), introduces a dual-certificate model, replaces conceptual registry integrity with a cryptographic Transparency Log, moves protocol adapters to a client SDK, and decouples discovery from the RA.

1.3. Relationship to Other Standards

HCS-14 [HCS14] uses DNS TXT records for agent discovery; an ANS Profile for HCS-14 allows any HCS-14 resolver to discover ANS agents. HCS-27 [HCS27] defines a checkpoint format for publishing the Transparency Log’s root hash to a distributed consensus topic. The IETF SCITT working group [I-D.ietf-scitt-architecture] defines an append-only transparency service; the ANS TL follows this model. DNS-AID [DNSAID] uses SVCB service binding records [RFC9460] for agent endpoint discovery; DNS-AID tells a client where to connect, ANS tells it whether to trust the agent.

1.4. Regulatory Context

Regulators are converging on verifiable agent identity as a compliance requirement. The EU AI Act (Article 50) requires transparency and identity disclosure for AI systems that interact with people. In the US, NIST’s Center for AI Standards and Innovation solicited public input on securing AI agent systems in early 2026. Both point toward mandatory verifiable agent identity.

1.5. Foundational Principles

Every agent maps to a unique, globally resolvable Fully Qualified Domain Name (FQDN). The ANSName format ans://v{version}.{agentHost} embeds the FQDN directly. The domain is the agent's permanent address; the version number changes, the domain does not.

Five design decisions follow from this foundation:

1. Domain control validation: The RA verifies ownership using the ACME protocol [RFC8555] (DNS-01 or HTTP-01 challenges) before attesting to any identity.

2. Decentralized discovery: The TL publishes sealed lifecycle events. Third-party Discovery Services subscribe, verify signatures, and build their own indexes.

3. Version-bound lifecycle: Every change to an agent's software or capabilities requires a new version number and a new registration.

4. Dual-certificate model: A Server Certificate from a public CA secures the stable FQDN. An Identity Certificate from a private CA attests to the version-bound ANSName.

5. Discoverable schemas: The ANS Trust Card links each protocol to a canonical URL. Schemas are public, versioned artifacts.

1.6. Registration Progression

The RA acts as a notary. It witnesses domain control, issues certificates, specifies DNS records, and seals the event into the TL. The minimum registration is intentionally low: a domain, a version, at least one endpoint, and an identity CSR.

Artifact	Required	Trust Index effect
Domain + version + endpoints + identity CSR	Yes	Baseline registration
OV or EV Server Certificate	No	Raises identity score
ANS Trust Card hosted at FQDN	No	Raises integrity score; enables content hash verification
Registration Metadata (agentCardContent)	No	Sealed hash enables integrity verification of the Trust Card
verifiableClaims in Trust Card	No	Feeds operational maturity signals (SOC 2, SBOM, compliance)
Principal binding (LEI, DID, biometric)	No	Raises identity score
SCITT receipt stapled to Trust Card	No	Enables offline TL verification; highest integrity signal

Table 1

1.7. Open Protocol Design

The protocol is designed for an open, federated market. Any organization can operate an RA, a TL, a Trust Index, or a Discovery Service. Multiple RAs can coexist, each issuing certificates and sealing events into its own TL. Multiple Trust Index providers can crawl the same logs and produce competing evaluations. Every interface is public. The DNS record types are standard. The TL verification API uses REST. The certificate model uses ACME, which any server can answer.

1.8. Conformance Roles

The protocol defines five roles:

Registration Authority: Validates domain control via ACME, issues

Identity Certificates from a Private CA, obtains or accepts Server Certificates from a Public CA, generates DNS record content, and submits lifecycle events to a TL.

Discovery Service: Consumes RA output and indexes agents. Any DNS provider, search engine, or marketplace can operate one.

Transparency Log operator: Seals signed events into an append-only cryptographic structure, signs checkpoints, and exposes a public verification API. A conforming TL operates as a SCITT Transparency Service [I-D.ietf-scitt-architecture].

Trust Index provider: Crawls TL events from one or more federated RAs and publishes evaluations as signed Verifiable Credentials.

Verifier: A client that checks an agent's identity before connecting. Verification is a client-side act, not a property the RA assigns. The verifier needs no relationship with the RA.

2. 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.

Agent Hosting Platform (AHP): The entity that hosts the agent's code and serves the live endpoints at the agent's FQDN.

Agent Integrity Monitor (AIM): A monitoring service that compares the live state of an agent's DNS records and Trust Card against the sealed TL records. It publishes findings but cannot command state changes.

ANSName: The canonical identifier for a registered agent, following the format ans://v{version}.{agentHost}.

ANS Trust Card: An optional COSE_Sign1 document hosted by the AHP containing protocol-native metadata augmented with ANS trust fields and a stapled SCITT receipt.

Discovery Service: An independent service that consumes RA output and indexes agents for searchability.

Identity Certificate: An X.509 certificate issued by the Private CA with a URI SAN containing the agent's ANSName.

Protocol Card: The protocol-native metadata file (A2A Agent Card, MCP manifest, OpenAPI document) created by the developer.

Registration Authority (RA): The trusted third party that validates domain control via ACME, issues certificates, generates DNS record content, and seals lifecycle events into the Transparency Log.

Registration Metadata: The agentCardContent field in the registration payload, translated from the Protocol Card by the SDK. The RA hashes it and seals the hash into the TL.

Server Certificate: An X.509 certificate issued by a public CA with a dNSName SAN for the agent's stable FQDN.

Transparency Log (TL): An append-only cryptographic data structure that seals signed events and provides inclusion and consistency proofs.

Trust Index: A scoring service that crawls TL events and external signals to produce a numeric trust evaluation for each agent.

3. Related Work

Traditional service discovery via DNS [RFC1035] provides name-to-address resolution but does not verify identity or track software versions. DNS-SD [RFC6763] adds local service discovery capabilities but does not address verifiable identity or complex capability matching on a global scale.

Research in multi-agent systems has explored various agent communication languages, such as those defined by FIPA. These works establish interaction protocols but do not address cryptographic identity verification or tamper-evident audit trails.

The SCITT architecture [I-D.ietf-scitt-architecture] generalizes transparency logs to arbitrary supply-chain statements with signed receipts. The ANS Transparency Log targets SCITT compliance so that its receipts are interoperable with other SCITT-compliant services.

W3C Decentralized Identifiers (DIDs) provide self-sovereign identity but lack built-in discovery, transparency, and domain-anchoring mechanisms. ANS complements DIDs: a DID can serve as a principal binding within an ANS registration.

4. Architecture

The architecture connects a central Registration Authority with Agent Hosting Platforms and shared internet infrastructure. Two chokepoints define trust flow: the RA, where identity enters the system, and the Transparency Log, where sealed evidence leaves it.

4.1. Registration Authority System

The RA system comprises:

Registration Authority (RA): Receives registration requests from AHPs, validates domain control via ACME [RFC8555], requests an Identity Certificate from the Private CA, obtains a Server Certificate from the Public CA (or accepts one the AHP brings), generates DNS records, and seals the registration into the TL.

Key Management System (KMS): Signs every TL checkpoint. If this key is compromised, every sealed record in the log becomes untrustworthy.

Provider Registry: Decouples an entity’s legal name from its identifier. When "AcmeCorp" becomes "MegaCorp," one record updates instead of re-registering thousands of agents.

Agent Integrity Monitor (AIM): Compares the live internet against what the RA sealed. When something does not match, it publishes a finding. It cannot revoke certificates or command state changes.

RA API: The AHP registers an agent, submits CSRs for both certificate types, and triggers ACME and DNS verification. Agent discovery flows from the Event Stream through independent indexing services, not through the RA.

4.2. Agent Hosting Platform System

The AHP hosts the agent’s code and serves the live endpoints at the agent’s FQDN. The AHP may also host an ANS Trust Card; the Trust Index rewards its presence. During registration, the AHP responds to ACME challenges so the RA can verify domain control, then receives both certificates and installs them in its keystore. The AHP provisions DNS records using content the RA generates. When the agent’s code changes, the AHP initiates a new version registration.

Interfaces hosted by the AHP include the Agent Functional Endpoint (the live service exposing capabilities) and the ANS Trust Card (the metadata document describing capabilities, endpoints, supported protocols, and ANS trust fields).

4.3. Transparency Log

The TL receives signed events from the RA, validates each signature, and seals the events into a cryptographic append-only structure. That structure produces two kinds of proof: inclusion proofs (proving a specific event exists in the log) and consistency proofs (proving the log has only grown, never shrunk or rewritten). The KMS signs each checkpoint. A conforming TL MUST operate as a SCITT Transparency Service [I-D.ietf-scitt-architecture], issuing binary COSE receipts as proof of inclusion.

Each event receives a sequence number that increases with every entry and never repeats. Events become visible only after the KMS signs a checkpoint, so a client querying the log always sees a consistent, finalized view. Each event carries a schema version so that field renames in future schemas do not break existing consumers.

A conforming TL MUST expose a REST API for external verifiers:

Endpoint	Purpose
GET /v1/agents/{agentId}	Sealed event, TL signature, and inclusion proof
GET /v1/agents/{agentId}/audit	Paginated history of all lifecycle events
GET /v1/log/checkpoint	Latest signed checkpoint: log size, root hash, KMS signature
GET /v1/log/checkpoint/history	Checkpoint history for consistency proof verification
GET /v1/log/schema/{version}	JSON Schema definition for a given event schema version
GET /root-keys	KMS verification keys, including historical keys

Table 2

The TL MUST distribute verification keys via /root-keys so that any verifier can check root signatures without contacting the RA. Verification MUST NOT require access to producer public keys, authentication for read-only operations, or knowledge of RA implementation details.

4.4. Event Stream

The Event Stream makes sealed events queryable. Every payload carries the RA's signature, verifiable against keys the RA registered at the TL. Consumers verify authenticity without contacting the TL. Discovery Services subscribe to the Event Stream to build searchable indexes.

4.5. Certificate Authorities

Two CAs, two trust roots, two revocation paths:

* Public CA: Issues Server Certificates. Revocation propagates through public OCSP/CRL [RFC6960].

* Private CA: Issues Identity Certificates on the RA's behalf. The RA requests issuance; the Private CA signs. The Private CA may be operated by the RA operator or by a separate organization. The Identity Certificate requires a URI SAN that no Public CA can issue. Only a Private CA, operating under its own issuance policy and certificate practice statement, can include the ans:// scheme. The Identity Certificate's validity period is not subject to CA/Browser Forum Baseline Requirements constraints.

4.6. DNS Provider

The external service that hosts the agent's DNS zone. The RA generates DNS record content; the AHP provisions it at the DNS provider.

4.7. Trust Framework: Three Layers

The RA answers one question: "Who are you?" It does not evaluate whether the agent is well-governed or well-behaved. Three layers of trust data together provide a complete answer.

Layer 1: Foundational identity (this protocol). The RA verifies domain control, issues certificates, and seals the Registration Metadata hash into the Transparency Log. This layer is the scope of this document.

Layer 2: Operational maturity (third-party attestors). SOC 2 reports, HIPAA compliance certificates, and SBOMs arrive as references in the Trust Card's verifiableClaims array. They update on the attestor's cycle: annually for an audit, quarterly for a compliance scan.

Layer 3: Behavioral reputation (real-time scoring). Transaction

records, settlement rates, and dispute flags. While these signals may go back months, they update in seconds.

No single source controls the evaluation. A companion Trust Index specification defines how a provider consumes sealed data from Layer 1 and external signals from Layers 2 and 3, computes a multi-dimensional trust evaluation, and returns it as a signed credential that clients use for authorization decisions.

4.8. The ANS SDK

The version-bound lifecycle requires a new registration on every code change. Without tooling, a developer who ships a one-line fix must also generate a key pair, build a CSR, submit it to the RA, wait for domain validation, install the new certificate, and update DNS. The SDK collapses that sequence into a single command.

The SDK contains a Protocol Adapter Layer that translates Protocol Cards (A2A Agent Cards, MCP manifests, OpenAPI documents) into the RA's registration payload format. This is a key architectural departure from the original ANS proposal [ANSv1], where protocol adapters resided within the RA. Moving adapters to the client side means the RA accepts a single payload format regardless of protocol, simplifying its implementation and reducing its attack surface.

The SDK also bundles a Trust Provisioner that manages the agent's trust store. In the single-RA phase, it installs the bootstrapping RA's Private CA root certificate. In the federated phase, it retrieves a trust bundle from the Federation Registry containing root certificates of all compliant RAs.

5. Data Model and Integrity

5.1. The ANSName

The canonical identifier for a registered agent has three components ([RFC1035], [RFC1123]):

```
ANSName = "ans://v" version "." agentHost
version = 1*DIGIT "." 1*DIGIT "." 1*DIGIT
agentHost = <FQDN per RFC 1035 and RFC 1123>
```

Example: ans://v1.0.0.sentiment-analyzer.example.com

Component	Example	Constraints
protocol	ans	Fixed. Always "ans".
version	1.0.0	Semantic version, numeric only: major.minor.patch. The "v" prefix is a literal part of the ANSName syntax, not part of the version string.
agentHost	sentiment-analyzer.example.com	FQDN per RFC 1035 and RFC 1123. MUST NOT exceed 237 octets.

Table 3

The 237-octet host limit: The RA derives DNS record names by prepending labels to the agentHost. The longest derived name is _acme-challenge.{agentHost}, consuming 17 octets (16 characters plus the label separator) of the 253-octet presentation-format domain name limit (RFC 1035 Section 2.3.4, RFC 1123 Section 2.1). 253 - 16 = 237.

The complete ANSName MUST NOT exceed 400 octets, providing headroom for safe transport across HTTP headers, TLS fields, and database columns.

Two metadata fields accompany the ANSName at registration but are not part of the identifier: agentDisplayName (required, max 64 characters, human-readable label for discovery UIs) and agentDescription (optional, max 150 characters).

5.2. Identifier Taxonomy

5.2.1. Registration Identifiers

Identifier	Assigned by	Mutable	Scope	Purpose
ANSName	Derived	No	Global	Which version is registered

				on which domain
FQDN	AHP	No	Global	Stable domain across all versions
Agent ID	RA	No	Issuing RA	Registration record's unique key
ProviderID	RA	No	Issuing RA	Who controls the domain
Supersedes ID	RA	No	Issuing RA	Previous version's record

Table 4

The ProviderID names the entity that controls the domain, not the entity that authored the software. For cross-RA correlation, the registration payload accepts an optional lei field (Legal Entity Identifier, ISO 17442).

When a platform registers an agent on behalf of a tenant, the ProviderID and lei field identify the platform and the tenant respectively. The tenant can close the delegation gap by issuing a W3C Verifiable Credential authorizing the platform to register on its behalf, included in the Trust Card's verifiableClaims array.

5.2.2. Transparency Log Identifiers

Log Entry ID (unique reference to each event), Sequence Number (monotonically increasing, enforces append-only property), Leaf Index (position in the cryptographic structure, used for inclusion proofs), and Tree Version (increments on KMS key rotation).

5.2.3. Reputation Continuity

The FQDN and Server Certificate together identify the agent's TLS endpoint across version bumps. When a registrant registers support.example.com at version 1.0.0, then bumps to 1.1.0, the domain and the Server Certificate stay the same. The ANSName resets. A Trust Index that accumulates reputation against the FQDN sees the full transaction history across both versions. The supersedes field in each TL event links the version chain.

Three mechanisms detect a change in control:

* ACME re-validation: The RA re-runs domain control validation at each renewal and version bump.

* RDAP monitoring: The AIM monitors the registrant entity in the registrar's RDAP response [RFC9083].

* Provider mismatch: When a new operator registers a version for an FQDN that already has an ACTIVE registration under a different ProviderID, the RA detects the conflict and MUST revoke the prior registration.

Upon detection of a control change, the RA MUST revoke the previous registration by sealing an AGENT_REVOKED event into the TL.

A principal binding -- an external, persistent identifier for the operating entity -- closes the gap between domain ownership and organizational identity. An LEI identifies the organization regardless of which domain it operates. When the lei field is present, a Trust Index can aggregate behavioral signals across all registrations sharing that LEI.

5.3. Three Agent Card Terms

1. Protocol Card: The protocol-native metadata file (A2A Agent Card, MCP manifest, OpenAPI document). Created by the developer.

2. Registration Metadata: The agentCardContent field in the registration payload. The SDK translates the Protocol Card into this format. The RA hashes it and seals the hash into the TL.

3. ANS Trust Card: An optional COSE_Sign1 document the AHP hosts at /.well-known/ans/trust-card.json. Contains protocol-native metadata augmented with ANS trust fields (ANSName, verifiable claims, trust references) and a stapled SCITT receipt. The AIM verifies the content hash against the value sealed when Registration Metadata is submitted.

5.4. Certificate Integrity

Certificate	Issued by	SAN type	Purpose
Server	Public CA	dNSName	Standard TLS for the stable FQDN
Identity	Private	URI SAN	Binds certificate to

		CA		a versioned ANSName

Table 5

The Identity Certificate requires a URI SAN. The ans:// scheme is syntactically valid per RFC 3986 [RFC3986] Section 3.1 and permitted in URI SANs per RFC 5280 [RFC5280] Section 4.2.1.6. CA/Browser Forum Baseline Requirements prohibit URI SANs in publicly trusted server certificates (BR Section 7.1.2.7.12). A Public CA cannot issue this certificate; only a Private CA can.

5.5. Agent State Lifecycle

The RA tracks registration state in its database. Only certain transitions produce TL events: entering ACTIVE, DEPRECATED, REVOKED, or EXPIRED, and renewal while ACTIVE (the AGENT_RENEWED event type). RENEWED is a TL event type, not a distinct registration state; the agent remains ACTIVE throughout the renewal process. Transitions before activation (PENDING to PENDING_DNS) are RA-internal.

State	Description
PENDING	Awaiting validation
PENDING_DNS	Domain validated; awaiting DNS record verification
ACTIVE	Operational
DEPRECATED	Signals consumers to migrate
REVOKED	Certificates invalidated (terminal)
EXPIRED	Requires renewal (terminal)

Table 6

External domains pass through PENDING_DNS; internal domains with automated DNS skip directly to ACTIVE. Revocation is idempotent. Cancellation before activation produces no TL event because the log-sealing step has not occurred.

5.6. Cryptographic Data Integrity Standards

Requirement	Standard	Rule
Canonicalization	JCS (RFC 8785)	All JSON MUST be canonicalized before signing or hashing
Signature format	JWS Detached	Payload is not embedded; signatures stored separately
Algorithm	ES256 (ECDSA P-256/SHA-256)	Default. MUST support algorithm agility.
Wire format	JWS Compact	<header>..signature (two dots; detached payload)

Table 7

Every signature MUST include protected headers: alg (signing algorithm), kid (key identifier), typ (type indicator), timestamp (Unix timestamp), and raId (RA instance identifier). All JSON payloads MUST be canonicalized per JCS [RFC8785] before signing. Signatures use JWS Compact Serialization [RFC7515] with the JSON data interchange format [RFC8259].

6. Trust, Security, and Attestation

6.1. Layered Trust and Verification Tiers

Trust rests on three independent verification dimensions (distinct from the three-layer trust data framework in Section 4.7, which separates data sources):

Layer	Mechanism	What it proves
Identity	Version-bound ANSName	Which version is registered on which domain
Cryptographic	KMS-signed checkpoint	The log has not been tampered with
Operational	raId per event	Which RA instance performed the validation

Table 8

A client verifies an agent through independent checks, each using a different trust channel:

Step	Check	What it proves	Trust channel
1	PKI certificate validation	Standard TLS	CA system
2	DANE record validation	Server Certificate fingerprint matches TLSA record	DNS (DNSSEC)
3	TL verification	Registration was sealed; tampered entries break the proof	TL (KMS-signed)

Table 9

Tier	Steps	Shorthand
Bronze	1	PKI only
Silver	1-2	PKI + DANE
Gold	1-3	PKI + DANE + TL

Table 10

A tier describes what the client verified, not a property the RA assigned. Two clients connecting to the same agent may reach different tiers depending on what checks they run.

TLSA parameters: The RA specifies TLSA 3 0 1 [sha256_hash]: DANE-EE (usage 3), full Server Certificate (selector 0), SHA-256 (matching type 1) [RFC6698]. Selector 0 produces the same hash as the badge fingerprint in the TL, so a single SHA-256 computation serves both DANE and badge verification.

DANE requires DNSSEC [RFC4033]. Per RFC 6698 Section 4, a TLSA RRset whose DNSSEC validation state is not "secure" MUST be treated as unusable, and the client falls back to standard PKIX certificate validation. The RA SHOULD verify DNSSEC status before provisioning TLSA records.

6.2. DNS Trust Anchor

The RA specifies one _ans-badge TXT record per ACTIVE version, pointing to that version’s badge in the TL. The AHP provisions it.

```
_ans-badge.{agentHost} IN TXT
  "v=ans-badge1; version=v1.0.0;
   url=https://{tl_host}/v1/agents/{agentId}"
```

Field	Required	Values
v	Yes	Always ans-badge1
version	Yes	Semver prefixed with v
url	Yes	Badge URL at the RA’s TL
ra	No	RA ID (reserved for federated deployments)

Table 11

When multiple versions are ACTIVE, each has its own _ans-badge record. A verifier that knows the version (from the Identity Certificate’s URI SAN) selects the matching record.

6.3. Cryptographic Verification Path

High-assurance verification walks this chain:

1. Verify the _ans-badge TXT record via DNSSEC.

2. Validate the JWS signature on the attestation badge using the RA's public key.

3. Verify that the event exists in the TL via an inclusion proof.

4. Validate the Signed Tree Head using the KMS key identifier.

5. Confirm the agent's current state matches the latest log entry.

6.4. Mutual TLS Between ANS-Registered Agents

The mTLS handshake proceeds as follows:

1. Caller sends ClientHello.

2. Server returns Server Certificate + CertificateRequest.

3. Caller presents its Identity Certificate.

4. Server verifies the caller's Identity Certificate against the ANS Private CA.

5. mTLS tunnel established. The caller knows the server's FQDN; the server knows the caller's ANSName.

6. Caller verifies the server's versioned identity via _ans-badge TXT record or TL query.

Steps 1-5 are standard mTLS. Step 6 is ANS-specific: the caller confirms the server's Server Certificate fingerprint matches the badge the RA sealed at registration.

When the agent's ANS Trust Card carries a stapled SCITT receipt, the caller verifies locally with no TL call. When the Trust Card is absent, the caller fetches the receipt from the _ans-badge URL. If the TL is unreachable, the caller falls back to PKI + DANE verification and records the downgrade.

When an agent acts on behalf of a human, the agent proves its own identity via mTLS. The human's authorization travels separately as a delegation token [RFC8693].

6.5. Key Management

The RA never generates, handles, or accesses an agent's private keys. The AHP owns its key lifecycle. The RA uses separate credentials for each external service integration, rotated on schedule.

Each RA instance MUST register at least one active public key with the TL before submitting events. Rotation uses an overlap window: new keys are registered with future validFrom dates; both old and new remain active during the transition. Producer private keys never leave the RA instance. Historical signatures remain valid after key expiration but not after revocation.

6.6. Channel vs. Message-Level Security

TLS secures transient, point-to-point API calls. Digital signatures secure durable artifacts that third parties or asynchronous consumers verify independently.

Payload	Signature	Verification
TL checkpoint	KMS key	Public, via /root-keys
RA attestation badge	RA key	Public, via RA's published key
Event producer signature	Producer key	Internal only; preserves chain of custody
Revocation requests	AHP key	Non-repudiation

Table 12

6.7. Privacy Considerations

Query privacy is out of scope for the RA. Discovery Services SHOULD implement privacy-preserving techniques (Private Information Retrieval, Anonymized Query Relays).

The TLS handshake reveals the agent's hostname to network observers. Encrypted Client Hello (ECH, RFC 9849) [RFC9849] encrypts the inner ClientHello, hiding the hostname and other connection parameters such as the ALPN list. When an AHP provides an ECH configuration during registration, the RA publishes it in the HTTPS record.

7. Operational Flow

The RA maintains a mutable database where registrations progress through states. The TL is append-only: once the RA seals an event, it cannot be altered. Before sealing, a registration is a draft in the RA's database. After sealing, the identity-bound fields are immutable. Every change requires a new version and a new TL event.

7.1. Initial Registration Flow

Registration has two stages:

1. AHP submits registration request.
2. RA validates domain control (ACME) [RFC8555].
3. RA obtains Server + Identity Certificates.
4. RA seals event into TL. (Point of no return.)
5. RA returns certificates + DNS record content to AHP.
6. AHP provisions DNS.

7.1.1. Stage 1: Pending Registration

The AHP submits a registration request containing:

* Identity: agentHost (FQDN), version (semver), agentDisplayName, optional agentDescription.
* Endpoints: Array of protocol-specific endpoints (minimum 1), each specifying protocol, agent URL, optional metadata URL, documentation URL, functions, and transports.
* Certificates: Identity CSR (required), optional Server CSR or existing Server Certificate (BYOC).
* Registration Metadata: Optional agentCardContent (translated from Protocol Card by SDK).
* Organization: Optional lei (Legal Entity Identifier).
* Privacy: Optional echConfigList (ECH configuration).

7.1.2. Stage 2: Activation

All validations must pass before activation:

Validation	Method
Domain control	ACME DNS-01 (RFC 8555 Section 8.4) or HTTP-01 (Section 8.3)
Organization identity	Separate OV-level process when applicable
Schema integrity	Fetch each metadataUrl, hash, compare
DNSSEC presence	Query for DNSKEY at agent’s zone. Advisory: warn if absent; do not block. Record as dnssecStatus in TL event.

Table 13

The activation sequence, irreversible once step 4 completes:

1. Certificate issuance: RA obtains the Identity Certificate and the Server Certificate from a Public CA (if CSR was provided) or validates a BYOC.

2. DNS record generation: RA generates record set content. AHP provisions the records.

3. Event payload: RA hashes Registration Metadata (if submitted) and assembles the event payload.

4. Log sealing: RA submits signed event to TL. Point of no return.

5. Artifact delivery: RA delivers certificates and DNS record content to AHP.

6. AHP provisions DNS: The AHP creates the DNS records using the content the RA specified.

7.2. Lifecycle Operations

Operation	Trigger	TL Event	DNS Effect
Version bump	Code/config change	AGENT_REGISTERED	New per-version records (_ans, _ans-badge); shared records unchanged

			(_443._tcp). AHP updates _ans-identity._tls.
Renewal	Certificate expiring, code unchanged	AGENT_RENEWED	None
Revocation	Agent shutdown or retirement	AGENT_REVOKED	Per-version removed; shared removed when last ACTIVE gone

Table 14

During a version bump, both old and new versions are ACTIVE. The Server Certificate stays active (tied to the FQDN, not the version). The RA does not impose a retirement timeline.

Rollbacks: The AHP deploys the old stable code as a new version (e.g., v1.0.2), registers it, and revokes the buggy version.

7.3. DNS Management Roles

Actor	Registration	Ongoing	Deregistration
AHP	Owns domain, manages A/AAAA	Autonomous DNS updates	Submits deregistration
RA	Generates ACME challenge, generates record content	Re-runs ACME, updates records	Deletes agent records
DNS provider	Hosts zone, processes API requests	Processes modifications	Processes deletions

Table 15

A CNAME at the agent's FQDN blocks HTTPS and SVCB records at the same label (RFC 1034 Section 3.6.2) but does not affect child-label records (_ans, _ans-badge, _443._tcp). When the RA detects a CNAME, it skips the HTTPS record. CNAME flattening by the DNS provider can mitigate this restriction.

7.4. DNS Record Set

The RA generates record content for child labels under the agent's FQDN. The AHP or domain owner provisions the records:

Record	Label	Type	Purpose	Per-ver.
Discovery	_ans.{host}	TXT	Protocol and metadata location	Yes
Badge	_ans-badge.{host}	TXT	TL badge URL for verification	Yes
DANE	_443._tcp.{host}	TLSA	Server Cert fingerprint	No
HTTPS	{host}	HTTPS	ALPN hints, ECH parameters	No
Identity DANE	_ans-identity._tls.{host}	TLSA	Identity Cert fingerprint	No

Table 16

7.4.1. The _ans DNS Record

The _ans TXT record is a connection hint published in DNS, telling a client which protocol the agent supports and where to find the metadata.

```
_ans.{agentHost} IN TXT
  "v=ans1; version=v1.0.0; p=a2a;
   url=https://agent.example.com/.well-known/agent-card.json"
```

Field	Required	Values
v	Yes	Always ans1
version	Yes	Semver prefixed with v
p	No	a2a, mcp, http. Omitted = any protocol.
url	Unless mode=direct	URL to the metadata file. Fallback: /.well-known/agent-card.json.
mode	Unless url present	card (default) or direct.

Table 17

Resolution logic: (1) Query all _ans TXT records for the FQDN. (2) Filter by p={target_protocol}. (3) Select the highest version (semver ordering). (4) mode=direct: connect to the FQDN; url present: fetch; neither: fall back to /.well-known/agent-card.json.

7.5. Coexistence with DNS-AID

A domain owner can publish DNS-AID SVCB records [DNSAID] [RFC9460] alongside _ans TXT records. The two record families occupy different DNS names and do not collide. An ANS-aware client reads _ans; a DNS-AID client reads the SVCB records under _agents.

7.6. Ongoing Integrity Verification

An AIM MUST distinguish between "Unreachable" (transient network failure, retry later) and "Mismatch" (the content has changed, remediation needed).

Check	What the AIM does	Pass condition
DNS pointer	Authoritative query for _ans and _ans-badge with DNSSEC	Records exist and validate
Trust Card integrity	Fetch ANS Trust Card from /.well-known/ans/trust-card.json, hash content	Hash matches sealed agentCardHash
Schema integrity	Fetch each schema.url, hash content	Hash matches schema.hash in Trust Card

Table 18

8. Interoperability and Standards Alignment

8.1. HCS-14 ANS Profile

Hiero’s HCS-14 standard [HCS14] independently arrived at DNS TXT records for agent discovery, using _agent.<nativeId> as the record name. An ANS Profile within HCS-14 allows resolvers to verify ANS DNS records, certificates, and log proofs through a standard interface without ANS-specific branching.

8.2. HCS-27 Merkle Tree Checkpoint

A companion specification [HCS27] defines a checkpoint format for publishing the TL’s root hash to a distributed consensus topic. The timestamp on the consensus topic is independently verifiable, strengthening the guarantee that historical log entries have not been backdated or rewritten.

8.3. IETF SCITT Alignment

The SCITT architecture [I-D.ietf-scitt-architecture] defines an append-only transparency service that accepts signed statements, registers them, and returns receipts. The ANS TL follows this model. A future goal is formal SCITT compliance, adopting COSE (RFC 9052) [RFC9052] signed statements and standardized receipt formats for interoperability with other SCITT-compliant transparency services.

8.4. Deployment Topologies

The same RA, TL, and certificate infrastructure can run at different scopes:

* Public ANS: The RA, TL, and event feeds are internet-facing. Any verifier that has installed the Private CA root can verify any agent's identity and audit its history.

* Internal ANS: An organization runs its own RA and TL behind its corporate network. Agents are visible only to participants on that network.

* Enterprise ANS as a service: A hosted internal ANS instance, operated on behalf of an enterprise in their own cloud account.

* Extranet ANS: A semi-private deployment shared among a defined set of partner organizations.

Each topology runs the same protocol. The difference is who can see the TL, who can query the event feeds, and who controls the Private CA.

9. Formal Properties

This section states the security properties the protocol specifies, with sufficient precision that a conforming implementation can be verified against them.

9.1. Two-Layer Proof Composition

A sealed event carries two independent proofs of existence, each anchored to a different trust root.

Layer 1 (Transparency Log): Let E be a lifecycle event sealed at leaf index i in a log of size n with Merkle root R. An inclusion proof consists of a hash path of length ceil(log2(n)). The proof is valid iff recomputing the root from LeafHash(E), the path, and i yields R, and the signature over R verifies against the TL's published KMS key. Implementations targeting SCITT compliance [I-D.ietf-scitt-architecture] encode the proof as a COSE_Sign1 receipt; the verification semantics are identical.

Layer 2 (Consensus Checkpoint): An HCS-27 checkpoint [HCS27] publishes R to a distributed consensus topic at timestamp T. The topic provides an independently verifiable total ordering.

Together, the two layers guarantee:

1. Inclusion: If a receipt for event E verifies against root R, then E was sealed into the log before the checkpoint containing R.

2. Non-repudiation: Removing E from the log after checkpoint publication requires producing a tree of size n′ >= n whose root R′ is consistent with R yet excludes leaf i. This contradicts the collision resistance of SHA-256.

3. Temporal binding: Because the consensus topic provides an independently verifiable timestamp, the TL operator cannot backdate an event.

An attacker who compromises the KMS alone can sign fraudulent roots, but cannot insert those roots into the consensus topic’s history. An attacker who compromises the consensus topic alone cannot produce valid inclusion proofs. Both channels must be compromised simultaneously to forge a sealed event with a credible timestamp.

9.2. Cryptographic Boundary Enforcement

The protocol distributes trust across four services. No service accepts another’s assertions without verifying a cryptographic signature.

Boundary	Verification	What compromise means
RA to TL	TL verifies producer signature (ES256, registered key) before sealing	A forged event without a valid producer key is rejected at ingestion
TL to Verifier	Verifier checks KMS signature on checkpoint via /root-keys	A tampered checkpoint fails signature validation at every client
TL to Event Stream	Each published event carries the producer's signature; consumers verify against keys registered at the TL	A fabricated event without a valid producer key fails signature check
AIM to RA	AIM reports findings; RA re-verifies each finding against TL-sealed records before acting	A false finding from a rogue monitor is detected on re-verification

Table 19

Compromise of any single component is detectable by the others. A compromised RA instance can sign fraudulent events, but every event records the instance's raId; an auditor isolates the damage. A compromised TL can sign fraudulent checkpoints, but cannot retroactively alter checkpoints already published to the HCS-27 consensus topic. A compromised AIM can file false reports, but the RA re-verifies each finding; the quorum requirement prevents a single monitor from triggering remediation. A compromised Event Stream can suppress or delay events, but cannot forge new ones.

This property does not hold against collusion between the RA and TL operators. HCS-27 consensus checkpoints and federation (Section 8.4) provide the primary mitigations.

10. Non-Functional Requirements

ID	Component	Threshold
NFR-A-01	RA	99.9% uptime
NFR-A-02	AIM	99.9% uptime

NFR-P-01a	RA	Activation processing < 120s median
NFR-P-02	TL	Sealing < 500ms median
NFR-P-03	Private CA	Revocation reflected in OCSP/CRL < 5 min
NFR-P-04	AIM	DNS change detection < 24h
NFR-P-05	TL	Batch processing < 5s
NFR-P-06	TL	Append: O(log n)
NFR-P-07	TL	Inclusion proof generation < 100ms
NFR-S-01	RA	1,000 registrations/hour
NFR-S-07	TL	Billions of events, sub-second append
NFR-C-02	TL	Standardized inclusion and consistency proof interfaces

Table 20

10.1. Failure Modes

Scenario	Consequence	RA Response
AHP unavailable	Identity Certificate expires. mTLS fails.	Detects expired status. Cannot auto-renew (AHP controls keys).
Domain expires	Endpoint unreachable. DCV fails.	Treats persistent DCV failure as security event and revokes registrations.
CA unavailable	New registrations blocked. Existing agents unaffected.	Queues pending requests. Reports dependency failure.

Table 21

11. Future Work

* Federated, multi-RA ecosystem: A marketplace of interoperable RAs, governed by a standards body analogous to the CA/Browser Forum.

* Persistent identifiers: On-chain IDs and DIDs (did:web, did:ethr) would provide reputation continuity across version bumps.

* Runtime integrity (ZKPs/TEEs): Zero-Knowledge Proofs and Trusted Execution Environment attestations would close the application integrity gap.

* SCITT formal compliance: Adopt COSE-based signed statements and standardized receipt formats.

* Verifiable claim types and issuer accreditation: Define specific claim type standards and accredit third-party issuers.

* Governance white paper: Detail name allocation, fee model, dispute arbitration, and root CA stewardship.

* Platform integration: Demonstrate integration with agent frameworks (LangChain, AutoGen, CrewAI) and cloud platforms.

12. Security Considerations

This section addresses threats identified through a MAESTRO framework analysis applied to the ANS architecture.

12.1. Agent Impersonation

Risk: An adversary impersonates a legitimate agent. Mitigation: Mandatory PKI with dual certificates. The Identity Certificate binds a specific code version to a verified domain. The agent must prove possession of the private key. Certificate chain validation against the Private CA root.

12.2. Registry Poisoning

Risk: An adversary injects malicious data (corrupted capabilities or endpoints) into the registry. Mitigation: The Transparency Log makes all sealed entries immutable and tamper-evident. The RA validates all payloads before sealing. The AIM continuously compares live DNS and Trust Card state against the TL’s sealed records.

12.3. Man-in-the-Middle

Risk: An adversary modifies communications between system components. Mitigation: Message integrity via digital signatures (JWS Detached). mTLS between ANS-registered agents using Identity Certificates. DANE provides a second trust channel independent of the CA system.

12.4. Denial of Service

Risk: Adversary incapacitates RA, TL, or resolution services. Mitigation: The data plane (mTLS handshakes between agents) continues during RA or AIM outages. Previously established trust proofs remain valid until certificates expire. Standard operational defenses (rate limiting, DDoS protection).

12.5. Transparency Log Compromise

Risk: An adversary modifies or deletes historical TL entries. Mitigation: Merkle tree structure makes tampering mathematically detectable via consistency proofs between any two tree states. KMS key separation ensures the signing key does not reside on the TL host. HCS-27 checkpoint anchoring to a public distributed ledger provides independently verifiable timestamps.

12.6. Compromised RA Instance

Risk: A single RA instance is breached and issues fraudulent registrations. Mitigation: Every TL entry records the raId of the processing instance. Auditors isolate every event that instance touched. Separation of duties requires the RA’s DNS permissions exclude TLSA write access, preventing a compromised RA from also forging DANE records.

12.7. SDK Supply Chain Compromise

Risk: The SDK manages key generation, CSR submission, and Trust Card assembly. A compromised SDK could exfiltrate private keys or submit altered registration payloads. Mitigation: The SDK is open-source and auditable. Key generation uses platform cryptography (HSM, TPM, or OS keystore when available). The RA validates every CSR and registration payload independently. Software signing and provenance attestation (e.g., SLSA, Sigstore) reduce the risk of tampered distributions.

12.8. Single Operator Risk

For a given registration, the same entity may operate both the RA and the TL that seals it. If that entity is compromised, it could forge events with no independent check. HCS-27 consensus checkpoints are the primary mitigation: the TL's root hash is anchored to an independently verifiable ledger that neither the RA nor the TL controls. Federation adds a second layer: an agent can register with an RA operated by one entity and have events sealed by a TL operated by another.

12.9. Ecosystem Integrity and Remediation

Four principles guard against malicious monitoring services:

1. Monitors report, the RA acts: External monitors publish findings. They cannot command state changes.

2. Reports are public and signed: Monitors publish to cryptographically signed feeds, building verifiable reputation.

3. Quorum before action: The RA MUST NOT act on a single report from one monitor.

4. Evidence must be verifiable: Every finding MUST include cryptographic proof. The RA re-verifies independently.

Suppression before revocation: When a finding meets the quorum requirement, the RA independently re-verifies the reported mismatch against the TL's sealed records. If confirmed, the RA publishes a suppression event to the TL. Discovery services that consume the event feed remove the agent from their indexes. Suppression is reversible; revocation is permanent.

13. IANA Considerations

This document has no IANA actions at this time.

Future revisions may request registration of the "ans" URI scheme with IANA per RFC 7595, and registration of underscore labels "_ans", "_ans-badge", and "_ans-identity" per RFC 8552.

14. References

14.1. Normative References

[RFC1035] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989, <https://www.rfc-editor.org/info/rfc1123>.

[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>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012, <https://www.rfc-editor.org/info/rfc6698>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, June 2013, <https://www.rfc-editor.org/info/rfc6960>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, May 2015, <https://www.rfc-editor.org/info/rfc7515>.

[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>.

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

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, March 2019, <https://www.rfc-editor.org/info/rfc8555>.

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

[RFC9052] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", RFC 9052, August 2022, <https://www.rfc-editor.org/info/rfc9052>.

[I-D.ietf-scitt-architecture] Birkholz, H., Delignat-Lavaud, A., Fournet, C., Deshpande, Y., and S. Lasker, "An Architecture for Trustworthy and Transparent Digital Supply Chains", Work in Progress, Internet-Draft, draft-ietf-scitt-architecture-22, October 2025, <https://datatracker.ietf.org/doc/draft-ietf-scitt-architecture/>.

14.2. Informative References

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, February 2013, <https://www.rfc-editor.org/info/rfc6763>.

[RFC8693] Jones, M., Nadalin, A., Campbell, B., Bradley, J., and C. Mortimore, "OAuth 2.0 Token Exchange", RFC 8693, January 2020, <https://www.rfc-editor.org/info/rfc8693>.

[RFC9083] Hollenbeck, S. and A. Newton, "JSON Responses for the Registration Data Access Protocol (RDAP)", RFC 9083, June 2021, <https://www.rfc-editor.org/info/rfc9083>.

[RFC9460] Schwartz, B., Bishop, M., and E. Nygren, "Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)", RFC 9460, November 2023, <https://www.rfc-editor.org/info/rfc9460>.

[RFC9849] Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", RFC 9849, 2025, <https://www.rfc-editor.org/info/rfc9849>.

[ANSv1] Narajala, V. S., Huang, K., Habler, I., and A. Sheriff, "Agent Name Service (ANS): A Universal Directory for Secure AI Agent Discovery and Interoperability", IEEE ICAIC 2026, 2025.

[MCP] Anthropic, "Model Context Protocol Specification", 2025, <https://modelcontextprotocol.io/specification>.

[A2A] Google, "Agent-to-Agent (A2A) Protocol", 2025, <https://google.github.io/A2A/>.

[HCS14] Hiero, "HCS-14: Universal Agent ID", 2025, <https://github.com/hashgraph/hedera-improvement-proposal/blob/main/HIP/hip-14.md>.

[HCS27] Hiero HCS-27 Working Group, "HCS-27: Merkle Tree Checkpoint Specification", 2025, <https://github.com/hashgraph/hedera-improvement-proposal/blob/main/HIP/hip-27.md>.

[DNSAID] Mozley, J., Williams, N., Sarikaya, B., and R. Schott, "DNS for AI Discovery", Work in Progress, Internet-Draft, draft-mozleywilliams-dnsop-dnsaid-01, March 2026, <https://datatracker.ietf.org/doc/draft-mozleywilliams-dnsop-dnsaid/>.

Appendix A. Data Structure Examples

All examples follow one agent, "Acme Support Agent" (support.example.com, version 1.5.0), through the registration lifecycle. CSRs and signatures are truncated for brevity.

A.1. Registration Request

The AHP submits this payload to the RA’s /register endpoint.

```
{
  "agentDisplayName": "Acme Support Agent",
  "agentDescription": "Customer support agent.",
  "version": "1.5.0",
  "agentHost": "support.example.com",
  "endpoints": [
    {
      "protocol": "A2A",
      "agentUrl": "wss://support.example.com/a2a",
      "metadataUrl":
        "https://support.example.com/.well-known/agent-card.json",
      "transports": ["STREAMABLE-HTTP", "SSE"],
      "functions": [
        { "id": "lookupOrder", "name": "Lookup Order",
          "tags": ["order", "support"] }
      ]
    },
    {
      "protocol": "MCP",
      "agentUrl": "https://support.example.com/mcp",
      "metadataUrl":
        "https://support.example.com/.well-known/mcp/
        server-card.json",
      "transports": ["STREAMABLE-HTTP"],
      "functions": [
        { "id": "getTicketStatus", "name": "Get Ticket Status",
          "tags": ["ticket", "support"] }
      ]
    }
  ],
  "identityCsrPEM": "...(truncated)...",
  "serverCsrPEM": "...(truncated)...",
  "lei": "549300EXAMPLE00LEI17",
  "agentCardContent": { "...see A.2..." }
}
```

A.2. ANS Trust Card

Hosted by the AHP at the URL advertised in the _ans DNS record. Built from the Protocol Card, augmented with ANS trust fields. This artifact is optional.

```
{
  "ansName": "ans://v1.5.0.support.example.com",
  "agentDisplayName": "Acme Support Agent",
  "version": "1.5.0",
  "agentHost": "support.example.com",
  "releaseChannel": "stable",
  "endpoints": [
    {
      "protocol": "A2A",
      "agentUrl": "wss://support.example.com/a2a",
      "metadataUrl":
        "https://support.example.com/.well-known/agent-card.json",
      "transports": ["STREAMABLE-HTTP", "SSE"],
      "functions": [
        { "id": "lookupOrder", "name": "Lookup Order",
          "tags": ["order", "support"] }
      ]
    }
  ],
  "securitySchemes": {
    "agentAuth": {
      "type": "mutual_tls",
      "description": "mTLS using the agent's Identity Certificate."
    }
  },
  "verifiableClaims": [
    {
      "type": "SOC2_TYPE2",
      "issuer": "did:web:auditor.example.com",
      "hash": "SHA256:9f3a2b...",
      "url":
        "https://auditor.example.com/reports/example-2026.json",
      "issuedAt": "2026-01-15T00:00:00Z",
      "expiresAt": "2027-01-15T00:00:00Z"
    },
    {
      "type": "SBOM_CYCLONEDX",
      "issuer": "self",
      "hash": "SHA256:4e7d1c...",
      "url":
        "https://support.example.com/.well-known/sbom.json",
      "issuedAt": "2026-02-01T00:00:00Z"
    }
  ]
}
```

A.3. TL Event Payload

```
{
  "ansId": "550e8400-e29b-41d4-a716-446655440000",
  "ansName": "ans://v1.5.0.support.example.com",
  "eventType": "AGENT_REGISTERED",
  "agent": {
    "host": "support.example.com",
    "name": "Acme Support Agent",
    "version": "v1.5.0",
    "providerId": "PID-8294",
    "lei": "549300EXAMPLE00LEI17"
  },
  "attestations": {
    "identityCert": {
      "fingerprint": "SHA256:22b8a804...",
      "type": "X509-OV-CLIENT"
    },
    "serverCert": {
      "fingerprint": "SHA256:d2b71bc0...",
      "type": "X509-DV-SERVER"
    },
    "dnsRecordsProvisioned": {
      "_ans": "v=ans1; version=v1.5.0; ...",
      "_ans-badge": "v=ans-badge1; version=v1.5.0; ..."
    },
    "domainValidation": "ACME-DNS-01",
    "dnssecStatus": "fully_validated"
  },
  "expiresAt": "2026-10-05T18:00:00.000000Z",
  "issuedAt": "2026-03-05T18:00:00.000000Z",
  "raId": "id-A",
  "timestamp": "2026-03-05T18:00:00.000000Z"
}
```


A.4. TL Badge Response

Badge consumers query GET /v1/agents/{agentId}.

```
{
  "schemaVersion": "V1",
  "status": "ACTIVE",
  "payload": {
    "logId": "550e8400-e29b-41d4-a716-446655440000",
    "producer": {
      "event": { "...same structure as A.3..." },
      "keyId": "id-B",
      "signature": "eyJhbGciOiJFUzI1NiJ9..."
    }
  },
  "signature": "eyJhbGciOiJFUzI1NiIsImtpZCI6...",
  "inclusionProof": {
    "leafHash": "abc123def456...",
    "leafIndex": 1234567,
    "treeSize": 9876543,
    "treeVersion": 1,
    "path": ["def456789abc...", "012345abcdef...", "..."],
    "rootHash": "current1234abcdef...",
    "rootSignature": "eyJhbGciOiJFUzI1NiJ9..."
  }
}
```

The path array contains log2(treeSize) hashes: about 30 for a billion events (under 1 KB).

A.5. DNS Records

```
$ORIGIN support.example.com.
$TTL 3600

; Core Location Records (Managed by AHP)
@       IN  A      192.0.2.50
@       IN  AAAA   2001:db8::50

; RA generates; AHP provisions
@            IN  HTTPS  1 . alpn="h2"
_443._tcp    IN  TLSA   3 0 1 <sha256_of_server_cert>

_ans         IN  TXT    "v=ans1; version=v1.5.0; p=a2a;
    url=https://support.example.com/.well-known/agent-card.json"
_ans         IN  TXT    "v=ans1; version=v1.5.0; p=mcp; mode=direct"

_ans-badge   IN  TXT    "v=ans-badge1; version=v1.5.0;
    url=https://{tl_host}/v1/agents/{agentId}"

; High-assurance (AHP-managed, per ADR 010)
_ans-identity._tls IN TLSA 3 0 1 <sha256_of_identity_cert>
```

A.6. AIM Failure Report

```
{
  "event_type": "integrity_failure_detected",
  "event_timestamp": "2026-03-06T11:00:00Z",
  "worker_id": "id-C",
  "agent_host": "support.example.com",
  "ans_name": "ans://v1.5.0.support.example.com",
  "check": {
    "record_type": "_ans",
    "failure_type": "MISMATCH",
    "expected_value": "v=ans1; version=v1.5.0; ...",
    "actual_value": "v=ans1; version=v1.5.0;
      url=https://malicious-site.example.com/evil-card.json"
  }
}
```

A.7. Revocation Request and Response

```
{
  "reason": "CESSATION_OF_OPERATION",
  "comments": "Service is being retired."
}
```

The RA returns the DNS records the AHP must delete:

```
{
  "agentId": "550e8400-e29b-41d4-a716-446655440000",
  "ansName": "ans://v1.5.0.support.example.com",
  "status": "REVOKED",
  "reason": "CESSATION_OF_OPERATION",
  "revokedAt": "2026-03-20T14:00:00Z",
  "dnsRecordsToRemove": [
    { "name": "support.example.com",
      "type": "HTTPS", "purpose": "DISCOVERY" },
    { "name": "_443._tcp.support.example.com",
      "type": "TLSA", "purpose": "CERTIFICATE_BINDING" },
    { "name": "_ans.support.example.com",
      "type": "TXT", "purpose": "TRUST" },
    { "name": "_ans-badge.support.example.com",
      "type": "TXT", "purpose": "BADGE" }
  ]
}
```

Authors' Addresses

Scott Courtney
GoDaddy

Email: scourtney@godaddy.com

Vineeth Sai Narajala
OWASP
Email: vineeth.sai@owasp.org

Ken Huang
DistributedApps.ai
Email: ken.huang@distributedapps.ai

Idan Habler
OWASP
Email: idan.habler@gmail.com

Akram Sheriff
Cisco Systems
Email: isheriff@cisco.com