dnsop
Internet-Draft
Intended status: Standards Track
Expires: 28 November 2026

J. Mozley
N. Williams
Infoblox, Inc.
B. Sarikaya
Unaffiliated
R. Schott
Deutsche Telekom
J. Damick
Amazon

27 May 2026

DNS for AI Discovery
draft-mozleywilliams-dnsop-dnsaid-02

Abstract

The document standardizes an approach for publishing AI agents in the Domain Name System (DNS) so that other agents can discover them. Discovery is then initiated based on one of three generic use cases, in increasing computational and latency cost: (1) the requestor knows both the organization and agent (2) the requestor knows the organization that provides a capability, but not the specific agent (3) the requestor knows the required capability, but not the organization or agent. Of these use cases only (1) and (2) are in scope for this document, although (3) can be derived from this specification.

DNS for AI Discovery (DNS-AID) is designed so that, once a client has learned an organization’s agents, subsequent transactions can utilize the first use case with the benefit of cacheable connectivity information that is learnable as an agentic skill. The mechanism uses Service Binding (SVCB) records for connectivity information and key meta data, a well known entry point using DNS-Based Service Discovery (DNS-SD) labels into an organization’s agent index, and optionally DNS Security Extensions (DNSSEC) and DNS-Based Authentication of Named Entities (DANE) TLSA records for trust and security. DNS-AID provides consumers of agent services with a direct connection method for agentic workloads not mediated by a third party. Organizations can use the same approach across public and private networks networks, providing consistency and common operational models, including publishing agents that are hosted in service provider domains.

This document introduces no new resource record types, opcodes, or response codes.

About This Document

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

The latest revision of this draft can be found at https://example.com/LATEST. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-mozleywilliams-dnsop-dnsaid/.

Discussion of this document takes place on the WG Working Group mailing list (mailto:WG@example.com), which is archived at https://example.com/WG.

Source for this draft and an issue tracker can be found at https://github.com/USER/REPO.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 28 November 2026.

Copyright Notice

Table of Contents

1. Introduction

This document standardizes an approach for publishing AI agents in the DNS so that other agents can discover them. Discovery can be initiated based on one of three use cases, where they are ranked in ascending order of latency and computational power required to complete:

1. The requestor knows both the agent and its origin domain, and a single DNS query provides connectivity information and key metadata (see Section 3.1)

2. The requestor knows the domain but not which agent provides a required capability, and an organization-level index returns a list of agents from which one is selected (see Section 3.2)

3. The requestor knows the required capability, but not the agent or domain, or knows neither, in which case it must rely on an external directory or search service (see Section 3.3)

Of the above discovery use cases, (1) and (2) are addressed by this document. The third use case can be derived from (2) based on a well-known entry point to an organization's index of agents, such that any organization could build a search service based on this information. In the event the capability is unknown, external search is likely used anyway, which is treated like (3).

1.1. Design

DNS-AID is designed so that, once a client has learned an organization's agent capabilities, subsequent queries can use the first use case. The intention is to provide a minimum DNS record set, that is cacheable or even learnable as an agentic skill, with each organization able to control the advertisement of its own agents. This also provides consumers of agent services with a direct connection method for agentic workloads not mediated by a third party. Organizations can use the same approach within internal networks, providing consistency across public and private networks. The mechanism uses SVCB and DNS-SD to include connectivity and metadata within a single DNS record, and optionally utilizes DANE TLSA records to support efficiently establishing encrypted communication.

A deployment using the DNS-AID approach uses the following:

* Service Binding (SVCB) records [RFC9460] for the agent endpoint and protocol

* TLSA records [RFC6698] MAY be used for the agent's TLS identity

* The records SHOULD be DNSSEC-signed [RFC9364] for data origin authentication and data integrity, if TLSA records are used they MUST be signed

* DNS-Based Service Discovery (DNS-SD) [RFC6763] MAY be used for compatibility with existing service-discovery tooling

* Several extensions - agent-to-zone control proofs, query-time policy signaling, and cross-domain search - are being explored in parallel and are summarized under Section 5.

We note in section Section 4 that there could be an option to fall back to TXT records, but this is not optimal given the utility and efficiency of the SVCB record type for service discovery.

1.2. Determinism

DNS discovery is deterministic and cacheable: a given query against a given zone returns the same SVCB and TLSA RRsets, signed and time-bounded, until the publisher rotates them. The downstream agent's decisions over the discovered metadata are not deterministic. Implementations and reviewers SHOULD treat DNS-AID as providing a verifiable transport for agent metadata, not as a guarantee that an agent will behave predictably given that metadata.

2. Conventions and Definitions

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 A program that exchanges structured messages with other programs (other agents, services, or human-facing applications), typically over an HTTP transport.

agent record The SVCB record set published at an agent's primary owner name, together with any AliasMode records that redirect to it.

agent protocol The application-layer protocol an agent speaks once a TLS connection has been established (for example, the Model Context Protocol or Agent-to-Agent). Carried in the bap SvcParamKey, or in alpn when only one protocol is supported.

capability descriptor A machine-readable document describing the operations, inputs, and outputs an agent exposes. The descriptor itself is fetched out of band; DNS-AID carries its URI and a SHA-256 digest of its canonical form.

organization index The list of agents an organization chooses to advertise as discoverable, published at _index._agents.<domain>.

primary owner The fully-qualified domain name at which an agent's

normative SVCB ServiceMode record is published.

3. Discovery Use Cases

This section of the document provides details of the three generic use cases, including DNS record usage.

3.1. Known Agent

This section covers the use case where a requestor knows both the agent and its origin domain.

A single SVCB record type query for agent-name.example.com returns IP addresses, transport, agent protocol, and capability metadata. A TLSA record type query authenticates the TLS endpoint, although implementations may use other mechanisms. This is the form publishers SHOULD support and the form requestors MUST try first.

The SVCB record of an agent can provide some or all of the following:

* TargetName -- The target of the SVCB record, which can be the same as the agent domain name or different in the cases such as where a service provider hosts an agent.

* ipv4hint and ipv6hint -- TargetName IP addressing.

* alpn -- A unique protocol suite used by the target (e.g. alpn=mcp,h2,h3) consisting of transport (e.g. h2, h3) and/or an application-layer (agent) protocol (e.g. mcp or a2a) see Section 3.1.1.

* bap -- An optional SvcParamKey for agent protocol (e.g. mcp, a2a). Carrying the agent protocol as a separate parameter lets consumers and policy engines match on agent protocol without parsing transport protocols in alpn. Publishers MAY place the agent protocol directly in alpn. This is considered experimental, see Section 5.1.

* well-known -- The well-known URI path as as described in [RFC8615] e.g. /.well-known/agent-card.json.

* cap -- Capability descriptor locator or inline identifier (e.g., a URN or compact JSON-Ref).

* cap-sha256 -- Capability base64url-encoded SHA-256 digest of the canonical capability descriptor.

* Optional policy and realm parameters for multi-tenant or compliance-scoped deployments, or operator and/or service-specific metadata. The exact syntax and registry policy for these keys are deferred to a future revision (see Section 5).

Operators MAY publish the same agent under an _agents.example.com inventory leaf (for organizations that prefer a single containing prefix for their agents) or under a canonical DNS-SD label (so off-the-shelf DNS-SD clients can enumerate). In these cases SVCB AliasMode MUST be used to point at the primary owner.

Examples of these records are shown below in Figure 1 and their usage in Figure 2:

```
# An agent named agent-name with connectivity and capability information
agent-name.example.com. 3600 IN SVCB 1 . (
    alpn="a2a"
    port=443
    ipv4hint=192.0.2.1
    ipv6hint=2001:db8::1
    well-known=agent-card.json
    cap=<capability descriptor locator>
    ...
)

# Agent agent-name where the TargetName is a hosted service
agent-name.example.com. 3600 IN SVCB 1 resource.service-provider.example (
    ...
)
```

Figure 1: SVCB Record Examples

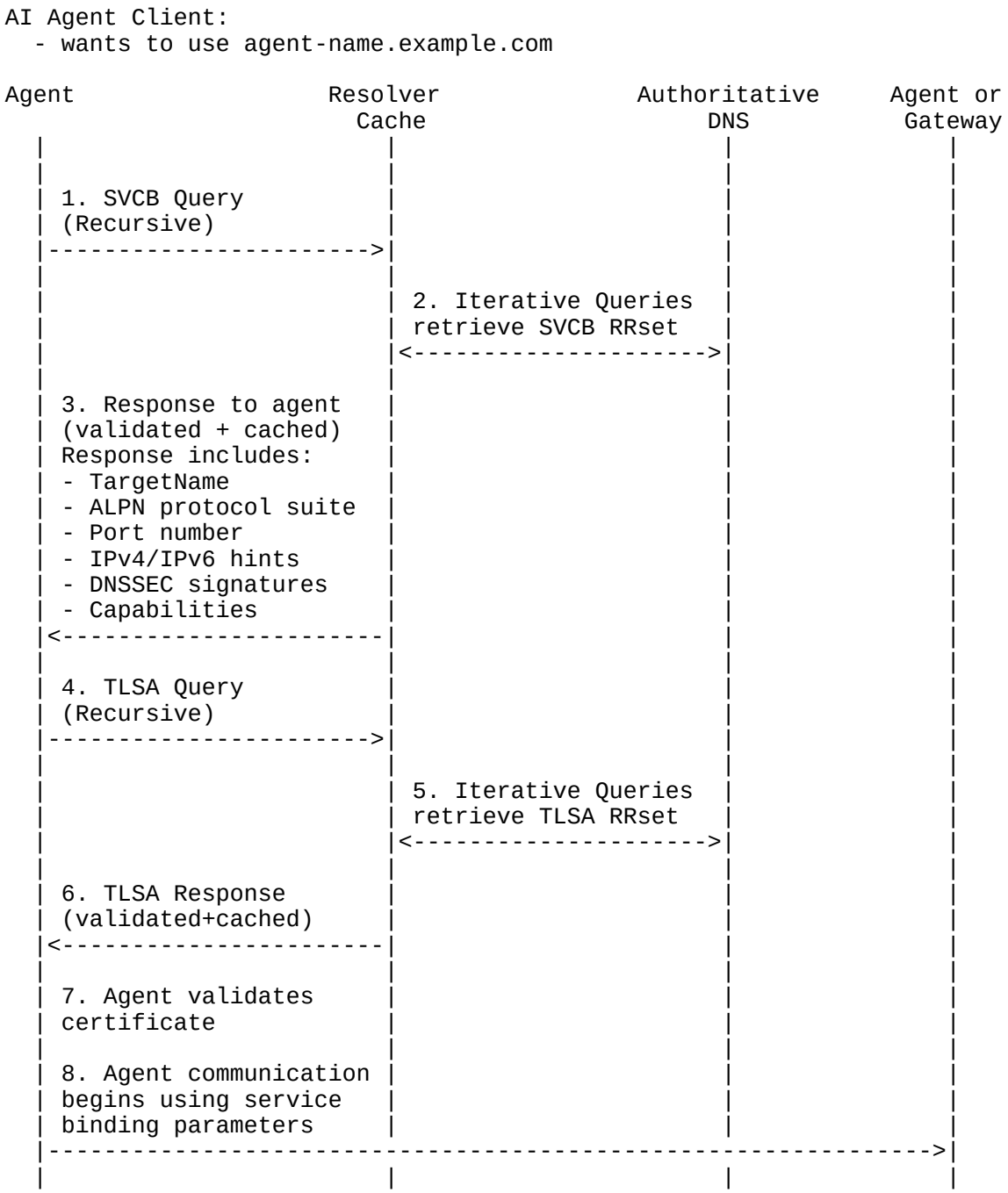


Figure 2: Discovery Example 1: Known Agent Discovery

3.1.1. Agents Supporting Multiple Protocols

This section details cases where the same agent might support multiple agent and transport protocols. [RFC9460], Section 7.1 describes the use of alpn SvcParamKeys in this regard for agent application-layer and transport protocols. Agents would process the RRset, and could choose from multiple SVCB records, selecting the preferred protocol suite.

Each application-layer (agent) protocol MUST be a separate record in an RRset e.g.

```
agent-name.example.com IN SVCB . alpn=mcp,h2,h3
agent-name.example.com IN SVCB . alpn=a2a,h2,h3
```

Multiple protocols MUST NOT be used in the alpn value as the SvcParamKey MUST uniquely identify a protocol suite:

```
agent-name.example.com IN SVCB . alpn=mcp,a2a,h2,h3 # MUST NOT be used
```

See [RFC9460], Section 7.1 for further details. Examples of records to support multiple protocols are shown below in Figure 3.

```
# An agent supporting multiple protocols
# Each record in an RRset is used a single application-layer (agent) protocol
agent-name.example.com IN SVCB . alpn=mcp,h2,h3
agent-name.example.com IN SVCB . alpn=a2a,h2

# TargetName can be different to support to support service providers
# or different endpoints
agent-name.example.com. 3600 IN SVCB 1 resource.service-provider.example (
    alpn=mcp,h2,h3
    well-known=/.well-known/agent-card.json
)
agent-name.example.com. 3600 IN SVCB 1 agent-name-a2a.example.com (
    alpn=a2a,h2
    well-known=/not-well-known/other-card.json
)

# It would be possible to use DNS-SD labels, although this is redundant
# as the protocol is in the alpn, but organizations might use this for
# compatibility with existing service-discovery tooling, or for
# filtering and security purposes
agent-name.example.com. 3600 IN SVCB 1 _a2a.agent-name.example.com (
    alpn=a2a,h2
)
```

Figure 3: SVCB Records for Multi-Protocol Support

3.2. Known Organization

This section covers the use case where a the requestor knows the organizational domain, but not which agent provides a required capability.

An SVCB record type query for _index._agents.example.com returns a pointer to an organization-specific registry of all agents. The requestor uses the data returned to select an agent and cache the selection so that the process in Section 3.1 can be used for subsequent interaction. The data provided at _index._agents.{domain}, protocols and schemas are out of scope for DNS-AID.

The TargetName MUST be used and MUST not contain underscores as used in DNS-SD labels, as public x.509 certificates will be used in communications with the index. Following the _index._agents.example.com SVCB record query, agents will use the TargetName for any TLSA record query.

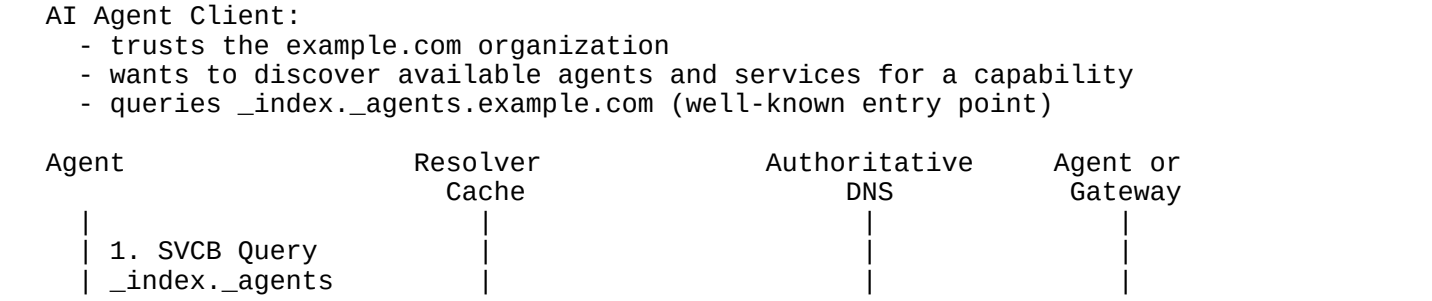
The labels _index._agents have been chosen as DNS-SD labels with the intent of registering them with IANA, as it is not possible to register a generic name such as agent-index.

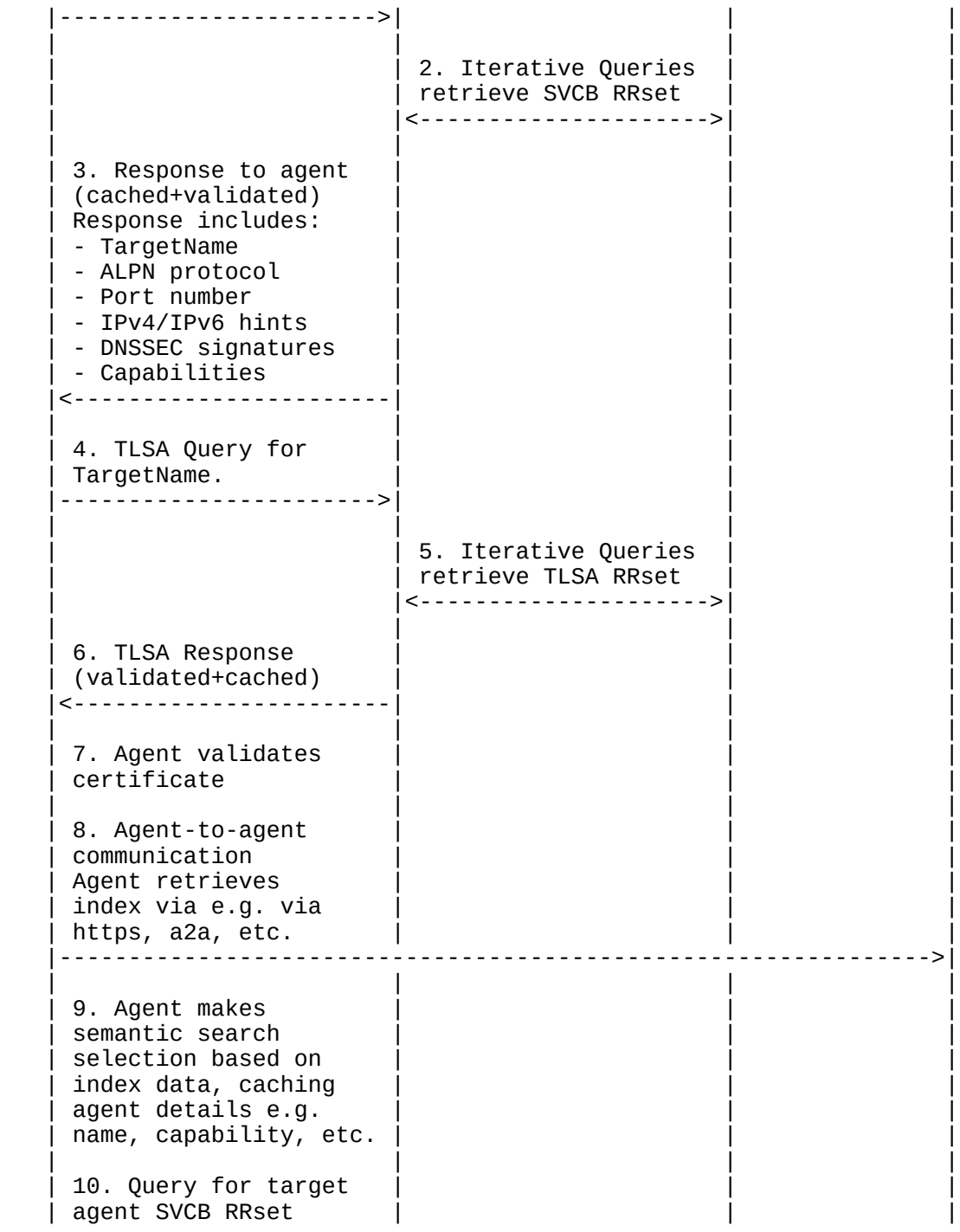
Examples of these records are shown below in Figure 4:

```
# Organizational agent index at a well-known name
_index._agents.example.com. 3600 IN SVCB 1 agent-index.example.com (
    ...
)

# Organizational agent index at a well-known name using a hosted service
_index._agents.example.com. 3600 IN SVCB 1 resource.service-provider.example (
    ...
)
```

Figure 4: Organizational Index Record Examples




2. Iterative Queries retrieve SVCB RRset
3. Response to agent (cached+validated) Response includes:
- TargetName
- ALPN protocol
- Port number
- IPv4/IPv6 hints
- DNSSEC signatures
- Capabilities
4. TLSA Query for TargetName.
5. Iterative Queries retrieve TLSA RRset
6. TLSA Response (validated+cached)
7. Agent validates certificate
8. Agent-to-agent communication Agent retrieves index via e.g. via https, a2a, etc.
9. Agent makes semantic search selection based on index data, caching agent details e.g. name, capability, etc.
10. Query for target agent SVCB RRset

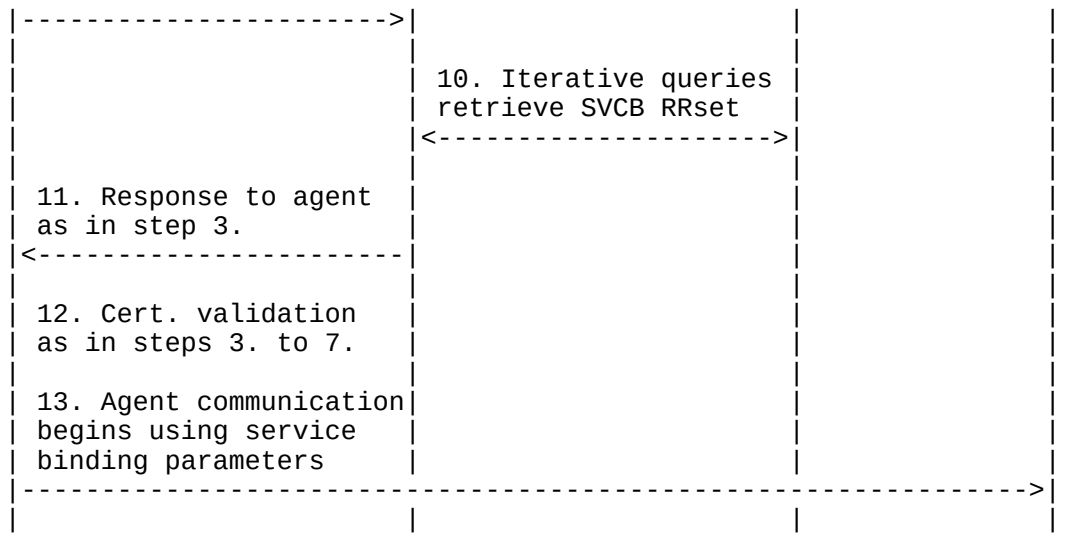


Figure 5: Discovery Status 2: Index-Based Discovery with Service Selection

3.3. Known Capability

This section covers the use case where the requestor knows the required capability, but not the organization or agent.

This is a _search_ based process and is out of the scope for DNS-AID. Directory services that aggregate records across organizations could be derived from Section 3.2 by querying _index._agents.{domain} and using that data for a search service.

4. Use of TXT Records as a Fallback From SVCB Records

TXT records could be used as a fallback for a lack of SVCB record support, likely by authoritative DNS management portals and services, as DNS server software has widespread support for this record type. Instead of using SVCB SvcParamKeys and values these could added to TXT record RDATA. This is not considered desirable as SVCB records are specifically design for service discovery, see [RFC9460]. One notable difference in using TXT records would be the lack of a TargetName that could be different from the domain name, allowing organizations to host services within a different domain, such as a service provider.

5. Future Work and Experimental Mechanisms

This section collects mechanisms that have been prototyped against the DNS-AID record set but that are not yet ready for normative specification in this document. Implementations MAY experiment with them; future revisions or separate documents are expected to standardize the ones that prove out. The headings below replace the briefer "Future Work & Unaddressed Portions" section in earlier revisions and consolidate the open questions raised there.

5.1. Bulk Agent Protocol

The bap parameter may be used experimentally to signal which version of agentic protocol to use, e.g. mcp=1.0, a2a=1.1. Pending hackathon attempts if it is a useful signal to include for efficient communications or additional noise / RDATA bloat.

5.2. Domain Control Validation

DNS-AID anticipates that agents and directories will sometimes need to prove that the publisher of an agent record controls the DNS zone in which it is published. The procedure in [I-D.draft-ietf-dnsop-domain-verification-techniques] is one candidate mechanism: the challenger issues a short-lived token that the claimant publishes at _agents-challenge.{domain} as TXT, and the challenger reads it back over a DNSSEC-validated path. A binding parameter such as bnd-req=svc:<agent>@<issuer> can scope a token to a specific agent leaf and issuer to prevent cross-challenger reuse.

A representative TXT record body and ABNF:

```
challenge   = token-attr 1*(SP attr)
token-attr  = "token=" 1*VCHAR
attr        = domain-attr / bnd-attr / expiry-attr
domain-attr = "domain=" 1*VCHAR
bnd-attr    = "bnd-req=svc:" 1*VCHAR "@" 1*VCHAR
expiry-attr = "expiry=" date-time   ; RFC 3339

_agents-challenge.acme.example. 60 IN TXT (
    "token=gjdqocfk4vhiq2bn" "domain=acme.example"
    "bnd-req=svc:api@acme.example" "expiry=2026-05-13T23:00:00Z" )
```

Open questions that should be answered before this is made normative: how a verifier scopes the number of records consulted for one verification, how expiry semantics interact with DNSSEC validity windows, and whether DCV results should be cached past their expiry. Consideration will also need to be given to the risk of this being used as an attack vector.

5.3. SVCB Indicating TLSA Support

An SvcParamKey in the agent SVCB record could be used to indicate TLSA support for the agent. As the SVCB query is the first one queried this might improve efficiency. Although the discovering agent may simply send both an SVCB and TLSA query simultaneously and infer support from the query responses, indicating the presence of a TLSA record would be more deterministic.

5.4. EDNS(0) Discovery Hints

A separate experimental mechanism allows a requestor to attach selector and metering hints to a DNS-AID query as an EDNS(0) option [RFC6891]. Substrate selectors (realm, transport, jurisdiction, minimum trust level) can influence the returned RRset and participate in cache keys; metering hints (intent class, freshness budget, parallelism) are advisory and do not fragment caches. A reference implementation and wire format are tracked in dns-aid-core (see docs/experimental/edns-signaling.md and the accompanying ABNF). No IANA option-code reservation is requested in this revision; the option uses the private-use code range.

5.5. Consolidated Registry and Cross-Domain Search

DNSAID-01 left open how a consolidated registry would operate -- whether it would scrape DNS-AID records published by individual organizations, or require attestation and verification from each publisher. The same question covers cross-domain _search_, i.e. locating agents whose origin domain the requestor does not know. This document treats both as upper-layer work; see [I-D.draft-narajala-courtney-ansv2] for one complementary proposal.

5.6. Operator and Service Metadata SvcParamKeys

The policy and realm SvcParamKeys are reserved for operator-controlled and service-controlled metadata that does not affect basic discovery: multi-tenant scoping, jurisdictional and compliance hints, and commercial signaling such as cost-per-unit-of-input, unit-of-input definitions, billing tiers, or rate-limit hints. The exact syntax of these payloads, the registry policy for any sub-keys that emerge, and the interaction between these parameters and the trust profile described in the Security Considerations section are open questions. Implementations are encouraged to experiment with concrete payloads in the private-use space and to report back so that a future revision can specify a common subset.

5.7. Connection Extensions as SvcParamKeys

Using connect-class and connect-meta provides a mechanism to signal which transport models to use, and with which pieces of metadata. Class may indicate fields like _direct_ [RFC9460] for hostname+port semantics and _lattice_ for integration with a vendor tools and services. Test implementations can use the Private Use parameter range and be moved to registered SvcParamKeys if they have wide spread applicability.

5.8. Zero Trust Extension SvcParamKeys

Adding a capability like enroll-uri may be relevant for mediated zero-trust overlay connections where a caller/service must first complete enrollment/authentication prior to invocation of any skills. This can be added via an SvcParamKey.

5.9. JSON-Encoded Organization Index

The TXT record-based form of _index._agents.{domain} may be suitable for small organizations and could be machine-parseable. A JSON-bodied variant, among others, has been discussed for larger inventories where TXT record chunking and multi-record assembly becomes complex and more subject to error. The encoding, signing, and validity-window rules for such a variant are open questions deferred to a future revision. It is anticipated that other standards will address organizational agent registries and the protocols to access them..

6. Security Considerations

This section discusses the security properties of DNS-AID itself; the security of the agent protocols in agent-to-agent communication is out of scope. The threat models in [RFC4033] (DNSSEC), [RFC9460], Section 10 (SVCB), and [RFC6698] (DANE) apply unchanged.

6.1. Authenticity, Integrity, and Trust

DNS-AID inherits authenticity from DNSSEC: a consumer that follows a validated chain of trust to a signed SVCB and TLSA record set has cryptographic evidence that those records were placed by an entity controlling the zone in which they were signed. DNS-AID does not, and cannot, assert that the agent reached through those records behaves benevolently. Authentic records may point at a malicious or compromised agent, and the capability descriptor referenced by the well-known parameter may itself contain instructions hostile to a consumer that interprets it (e.g. prompt-injection attacks against language-model agents). Consumers MUST treat the records published

in DNS-AID as a verifiable transport for metadata, not as a trust signal in their own right; trust judgments MUST be made out of band by combining DNS-AID records with reputation, attestation, or organizational policy systems.

The cap-sha256 SvcParamKey provides integrity for the capability descriptor: a consumer that fetches the descriptor from the URI carried in well-known and finds that its SHA-256 digest does not match the value in cap-sha256 MUST refuse to use it. cap-sha256 is an integrity check, not a trust signal.

6.2. TLS Endpoint Authentication

DNS-AID consumers SHOULD authenticate the TLS endpoint advertised by an SVCB record using DANE TLSA records [RFC6698] [RFC7671] at the conventional _443._tcp.<owner> name. Three operational postures are common:

permissive (default for general-internet consumers) TLSA is queried; if a matching record is present and matches the presented certificate, the connection is pinned to it. If no TLSA record is present, the consumer falls back to WebPKI validation.

preferred As permissive, but the absence of a TLSA record is noted in the consumer’s audit log.

strict A connection is refused unless a TLSA record is present and matches.

This document does not mandate any one posture. Strict-by-default is not appropriate for the general Internet today given current DANE adoption; permissive-with-fallback matches the implementation experience of [DNS-AID-CORE].

6.3. Downgrade Resistance

A publisher that requires a consumer to honor a particular SvcParamKey SHOULD list that key in the SVCB mandatory= parameter per [RFC9460], Section 8. A consumer that does not implement a key listed in mandatory= MUST skip the record. This prevents an attacker on the path from substituting an older subset of parameters without breaking discovery for old consumers.

6.4. DNSSEC Dependency and Operational Pitfalls

DNS-AID is deployable without DNSSEC, but the authenticity guarantees it offers depend on a validated path from the root to the zone of interest. Consumers SHOULD validate DNSSEC and SHOULD refuse to act on bogus or unverifiable records. Zone operators SHOULD follow the operational guidance in [RFC6781] and SHOULD avoid signing algorithms or key sizes that resolvers in the target ecosystem are likely to ignore.

The mandatory= mechanism and DNSSEC validation interact: an unsigned record substituted in transit may cause a consumer to fall back to a less safe code path. Consumers MUST NOT relax DNSSEC validation requirements based on the apparent absence of mandatory= keys.

6.5. Threat-Model Cross-References

The threat catalogs maintained by the OWASP Multi-Agentic System Threat Modeling work [OWASP-MAESTRO] cover many agent-layer concerns that DNS-AID does not address directly. The substrate-level threats that DNS-AID does address (transport-identity spoofing, capability poisoning, downgrade) map roughly onto MAESTRO threats T47 / T7.1 / T9 (rogue server, agent impersonation), BV-2 (tool-description poisoning), and T7.6 (transport fallback downgrade). DNS-AID does not address workload-level MAESTRO threats; consumers SHOULD treat DNS-AID as one layer in defense in depth.

6.6. Privacy Considerations

The names queried during DNS-AID discovery are visible to recursive resolvers and to any on-path observer of unencrypted DNS traffic. Consumers SHOULD use DNS over TLS [RFC7858] or DNS over HTTPS [RFC8484] where available. EDNS Client Subnet [RFC7871] SHOULD NOT be sent on DNS-AID queries that would disclose user-identifying agent names; operators SHOULD set ECS scope to zero where DNS-AID is the only consumer of a zone.

7. IANA Considerations

This document requests several IANA registrations to support DNS-AID. Final selection of code points, names, and registration policies depends on the outcome of working-group discussion and IESG review. The templates below follow the conventions used by [RFC9460] (SVCB), [RFC8552] (underscored DNS node names), and [RFC7301] (ALPN).

7.1. DNS Service Parameter Keys

IANA is requested to register the following entries in the "Service Parameter Keys (SvcParamKeys)" registry established by [RFC9460], Section 14.3.2. The numeric SvcParamKey values are deferred to IANA assignment; the names below are the strings carried in zonefile presentation form per [RFC9460], Section 2.1.

SvcParamKey	Meaning	Reference
cap	capability descriptor locator or inline identifier (e.g., a URN or compact JSON-Ref)	this document
cap-sha256	capability base64url-encoded SHA-256 digest of the canonical capability descriptor	this document
policy	URI of an associated policy bundle	this document
realm	opaque token for multi-tenant scoping or authz realm selection during protocol bootstrapping	this document
well-known	Well-Known Uniform Resource Identifiers [RFC8615] where the .well-known can be assumed, so the value of this key could be agent-card.json	this document
bap	Bulk agent protocols supported at this endpoint	this document

Table 1

For each entry, the change controller is "IETF" and the status is "permanent". The intended registration policy is Standards Action per [RFC9460], Section 14.3.2. The syntactic format for each value is described inline with the parameter in the Introduction; a separate Customization section may be added in a future revision once values stabilize.

7.2. Underscored DNS Node Names

IANA is requested to register the following entries in the "Underscored and Globally Scoped DNS Node Names" registry established by [RFC8552], Section 4.

RR Type	_NODE NAME	Reference
SVCB	_agents	this document
SVCB	_index	this document
TXT	_agents-challenge	this document (experimental; see Section 5.2)

Table 2

The _agents-challenge registration is requested but tied to an experimental mechanism (see Section 5.2); a future revision is expected to confirm or relinquish it.

7.3. Application-Layer Protocol Negotiation (ALPN) Protocol IDs

If publishers choose to advertise agent protocols directly in the alpn SvcParamKey (an option permitted when only one protocol suite is supported), IANA is requested to register the following entries in the "TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs" registry per [RFC7301]:

Protocol	Identification Sequence	Reference
Model Context Protocol	mcp	this Document
Agent-to-Agent	a2a	this Document

Table 3

The exact spelling of each identifier is to be confirmed with the maintainers of the respective protocols; the entries above are placeholders. ALPN registration policy is Specification Required per [RFC7301].

8. References

8.1. 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/rfc/rfc2119>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/rfc/rfc4033>.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <https://www.rfc-editor.org/rfc/rfc6698>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/rfc/rfc7301>.

[RFC7671] Dukhovni, V. and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance", RFC 7671, DOI 10.17487/RFC7671, October 2015, <https://www.rfc-editor.org/rfc/rfc7671>.

[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/rfc/rfc8174>.

[RFC8552] Crocker, D., "Scoped Interpretation of DNS Resource Records through "Underscored" Naming of Attribute Leaves", BCP 222, RFC 8552, DOI 10.17487/RFC8552, March 2019, <https://www.rfc-editor.org/rfc/rfc8552>.

[RFC9364] Hoffman, P., "DNS Security Extensions (DNSSEC)", BCP 237, RFC 9364, DOI 10.17487/RFC9364, February 2023, <https://www.rfc-editor.org/rfc/rfc9364>.

[RFC9460] Schwartz, B., Bishop, M., and E. Nygren, "Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)", RFC 9460, DOI 10.17487/RFC9460, November 2023, <https://www.rfc-editor.org/rfc/rfc9460>.

8.2. Informative References

[DNS-AID-CORE]
Infoblox, Inc. (contribution to Linux Foundation pending), "dns-aid-core: Reference implementation of DNS-AID", 2026, <https://github.com/infobloxopen/dns-aid-core>.

[I-D.draft-ietf-dnsop-domain-verification-techniques]
Sahib, S. K., Huque, S., Wouters, P., Nygren, E., and T. Wicinski, "Domain Control Validation using DNS", Work in Progress, Internet-Draft, draft-ietf-dnsop-domain-verification-techniques-12, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-domain-verification-techniques-12>.

[I-D.draft-narajala-courtney-ansv2]
Courtney, S., Narajala, V. S., Huang, K., Habler, I., and A. Sheriff, "Agent Name Service v2 (ANS): A Domain-Anchored Trust Layer for Autonomous AI Agent Identity", Work in Progress, Internet-Draft, draft-narajala-courtney-ansv2-01, 13 April 2026, <https://datatracker.ietf.org/doc/html/draft-narajala-courtney-ansv2-01>.

[OWASP-MAESTRO]
OWASP GenAI Security Project, "Multi-Agentic System Threat Modeling Guide v1.0", April 2025, <https://genai.owasp.org/resource/multi-agentic-system-threat-modeling-guide-v1-0/>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <https://www.rfc-editor.org/rfc/rfc6763>.

[RFC6781] Kolkman, O., Mekking, W., and R. Gieben, "DNSSEC Operational Practices, Version 2", RFC 6781, DOI 10.17487/RFC6781, December 2012, <https://www.rfc-editor.org/rfc/rfc6781>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/rfc/rfc6891>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/rfc/rfc7858>.

[RFC7871] Contavalli, C., van der Gaast, W., Lawrence, D., and W. Kumari, "Client Subnet in DNS Queries", RFC 7871, DOI 10.17487/RFC7871, May 2016, <https://www.rfc-editor.org/rfc/rfc7871>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/rfc/rfc8484>.

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

Contributors

The authors thank the following people for their substantial input to this document: Scott Courtney (GoDaddy), Ralf Weber (Akamai), and John Zinky (Akamai).

Acknowledgments

The authors also would like to thank Connor Snitker (GoDaddy), Chungwei Yen (GoDaddy), Patrick Mevzek (GoDaddy), Jim Gilbert (Akamai), Hema Seshadri (Akamai), Aaron Parecki (Okta), Ihab Shraim (CSC), Vincent DAngelo (CSC), and Nick Sullivan for their feedback and discussion.

Authors' Addresses

Jim Mozley
Infoblox, Inc.
Email: jmozley@infoblox.com

Nic Williams
Infoblox, Inc.
Email: nic@infoblox.com

Behcet Sarikaya
Unaffiliated
Email: sarikaya@ieee.org

Roland Schott
Deutsche Telekom
Email: roland.schott@telekom.de

Jeff Damick
Amazon
Email: jdamick@amazon.com