dnsop
Internet-Draft
Intended status: Standards Track
Expires: 19 April 2026

J. Mozley
N. Williams
Infoblox, Inc.
B. Sarikaya
Unaffiliated
R. Schott
Deutsche Telecom
16 October 2025

Brokered Agent Network for DNS AI Discovery
draft-mozleywilliams-dnsop-bandaid-00

Abstract

The emerging field of agent-to-agent protocol standards introduces new requirements in order to facilitate discovery, trust signaling, session negotiation and communication. This document specifies a method for utilizing the Domain Name System (DNS) to facilitate scalable and interoperable discovery between AI agents. The proposed mechanism, referred to as _Brokered Agent Network for DNS AI Discovery_ (BANDAID), defines a structured DNS namespace and record usage model to support metadata exchange and capability advertisement.

BANDAID introduces a leaf zone convention (e.g., _agents.example.com) containing Service Binding (SVCB) records (e.g., chat._agents.example.com) that encode application-specific metadata. These records enable agents to retrieve operational parameters prior to initiating a session, supporting both targeted lookups and capability-based discovery. The approach leverages existing DNS infrastructure, including DNS Service Discovery (DNS-SD), DNSSEC, and DANE, to provide integrity, authenticity, and automation without requiring human intervention. Lastly, the draft for Domain Control Validation (DCV) proposes a best current practice to prove an agent is authorized to act on behalf of a domain.

This document proposes no change to the structure of DNS messages, and no new operation codes, response codes, resource record types, or any other new DNS protocol values.

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-bandaid/.

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 19 April 2026.

Copyright Notice

Table of Contents

1. Conventions and Definitions 4
2. Introduction 4

2.1. Challenges with Existing Proposed Mechanisms and Benefits of DNS 5
2.1.1. Development 6
2.1.2. Opportunity Cost 6
2.1.3. Governance 7
2.1.4. Sovereignty 8
2.1.5. Commerce 8
2.1.6. Application 9
2.1.7. Granularity 9
2.1.8. Performance 10
2.1.9. Lifecycle / Management 10
2.1.10. Trust 11
2.1.11. Agent Properties 11
2.1.12. Control 12
2.2. Mechanisms for Successful Agent Communication 13
2.2.1. Service Binding Records, aka SVCB 13
2.2.2. DNS Security Extensions, aka DNSSEC 14
2.2.3. DNS Service Discovery, aka DNS-SD 15
2.2.4. DNS-Based Authentication of Named Entities, aka DANE 15
2.2.5. Domain Control Validation, aka DCV 16
2.2.6. Service Resilience 17
2.2.7. Observability, Auditability 17
2.2.8. Interoperability 18
2.2.9. Multitenancy, Federation 18
2.3. Architecture 18
2.3.1. Example Communication 18
2.3.2. Example Record 18
2.3.3. Example Use Cases 19
3. BANDAID Overview 20
3.1. Zone and Other Requirements 20
3.1.1. Delegation and Chain of Trust 20
3.1.2. Algorithms and Parameters 21
3.1.3. Authenticated Denial of Existence 21
3.1.4. Signature Validity 21
3.1.5. Key Rollover 21
3.1.6. Transport and Message Size 22
3.1.7. Transfer and Integrity 22
3.1.8. Monitoring 22
3.1.9. Abuse Resistance 22
3.1.10. BANDAID Records 22
3.1.11. Performance Optimization(s) 23
3.1.12. Customization(s) 24
4. Implementation Guidance 25
4.1. AI Operators 25
4.1.1. Separation of Roles 25
4.1.2. Local Root Zone Copy 25
4.1.3. Aggressive Caching and Prefetch 25

4.1.4. EDNS(0) Transport Resilience 26
4.1.5. Load Distribution 26
4.1.6. Monitoring and Telemetry 26
4.1.7. Encrypted Resolver Steering (Informative) 27
4.2. AI Consumers 27
4.2.1. DNS-Based Service Discovery 27
4.2.2. Communication Failure / Retry 27
4.2.3. Domain Control Validation 28
4.3. AI Developers 28
4.3.1. Publishing Schema 28
4.3.2. TTLs, Update Agility 28
4.3.3. Capability Retrieval 29
4.3.4. Example DCV Flow 29
4.3.5. Example Zonefile 29
5. Conclusion 32
5.1. Future Work & Unaddressed Portions 32
6. Security Considerations 33
7. IANA Considerations 33
8. References 33
8.1. Normative References 33
8.2. Informative References 33
Acknowledgments 37
Authors' Addresses 37

1. 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.

2. Introduction

Agent-to-agent communication introduces novel requirements for service discovery, authorization signaling, and metadata exchange. While several proposals have explored new discovery protocols, the DNS remains the most widely deployed and interoperable mechanism for service discovery, with decades of operational maturity. Existing DNS record types (including SRV, NAPTR, SVCB, TXT, and A/AAAA) are routinely used to locate services and convey configuration metadata. It is therefore reasonable to assume that AI agents will interact with this network protocol during discovery, session initiation, and their application exchange.

If a user at $company asks their agent "clean up my Salesforce data based on this email," how does $company agent then know where to find Salesforce's agents? How would the Salesforce agent validate the

$company agent and make the correct requested changes to Salesforce data? The former question is a DNS-solvable problem, the latter is not in the scope of this draft.

DNS can convey authorization and intent: DNS provides an organization a way to indicate that an agent is authorized to act on behalf of a domain (DCV draft), publish allowed endpoints or capabilities (SVCB), advertise service metadata in a way that' s cacheable, verifiable, and language-agnostic (tree domain etc), all while creating tamper-proof records and binding identities to certificates (DNSSEC + DANE). Additionally, DNS contributes to agent security by enabling connections to trusted nameservers, mitigating risks from typosquatted or malicious domains, and enforcing geographic or jurisdictional boundaries through country-code top-level domains (ccTLDs) and regionally scoped authoritative servers. These features collectively support secure, policy-compliant agent interactions at scale.

Unlike proposals that rely on centralized registries or ungoverned discovery networks, BANDAID builds upon existing infrastructure and governance models. It preserves operator sovereignty, supports decentralized publication, and benefits from support in the form of legal dispute processes and vulnerability disclosure programs.

This document aims to address key concerns in governance, commerce, security, and performance, allowing agent protocol designers to focus on data exchange standards while relying on the DNS for discovery. Agent-to-agent communication would take place after initial discovery, but some key data can be put into the DNS as a shortcut to make it more efficient.

The problem statement addressed by this draft is [I-D.draft-mozley-aidiscovery] and also relevant is [I-D.draft-rosenberg-ai-protocols] although this draft is related to the discovery of agents rather than protocols used directly in agent-to-agent communication.

2.1. Challenges with Existing Proposed Mechanisms and Benefits of DNS

Several alternative approaches to agent discovery have been proposed, including centralized registries, blockchain-based naming systems, and proprietary service directories. While these models may offer novel features, they introduce significant architectural, operational, and governance challenges. Centralized registries concentrate control and introduce single points of failure, limiting resilience and scalability.

Blockchain-based systems often lack integration with existing internet infrastructure, suffer from latency and cost constraints, and present jurisdictional ambiguity due to decentralized governance. Proprietary directories fragment the discovery ecosystem and inhibit interoperability across agent frameworks and administrative domains. Furthermore, securely delegating authority to AI solution providers on behalf of your organization requires interfacing between this central authority as opposed to business-to-business.

2.1.1. Development

The DNS benefits from decades of global development and operational deployment. It is supported by mature ecosystems for vulnerability disclosure, patch management, and protocol evolution, with active participation from security researchers, vendors, and operators worldwide. This distributed development model ensures that no single entity maintains disproportionate control over the protocol's trajectory or security posture.

DNS infrastructure is also tightly integrated with existing cybersecurity practices. Connections to trusted nameservers, protections against typosquatted or malicious domains, and support for jurisdictional scoping via country-code top-level domains (ccTLDs) contribute to a robust security baseline. These features are particularly relevant in agent-to-agent contexts, where automated systems make trust decisions without human oversight... and importantly, may be maliciously modified without human awareness.

In contrast, the introduction of a novel discovery protocol would require the establishment of new governance structures, security models, and operational tooling. Early implementations would likely be centralized among a limited set of stakeholders, introducing bottlenecks in protocol evolution and risk mitigation. Building a comparable ecosystem would require sustained investment, cross-organizational coordination, and long-term trust-building—resources that DNS already possesses.

2.1.2. Opportunity Cost

Introducing a new protocol for agent-to-agent discovery imposes nontrivial infrastructure and operational requirements. In particular, organizations may be required to deploy and maintain additional components such as specialized cache servers or registry interfaces, which increases complexity and operational overhead. Some proposals suggest a generic top level domain (gTLD) like .aiagent, which is more cost to advertise as domains are purchased by early opportunists. These requirements may present barriers to adoption, especially for small or resource-constrained entities, and

risk reinforcing disparities in access to AI technologies.

DNS, by contrast, is already deeply integrated into global infrastructure and supported by mature tooling, operational expertise, and deployment models. Its use for agent discovery enables rapid adoption without introducing new layers of technical debt. Leveraging DNS allows organizations to participate in agent-to-agent ecosystems using existing infrastructure, reducing onboarding friction and accelerating innovation. This approach supports incremental deployment and minimizes opportunity cost by avoiding the need for parallel protocol development and infrastructure investment.

2.1.3. Governance

The DNS operates within a well-established and internationally recognized governance framework. Oversight is provided by entities such as the Internet Assigned Numbers Authority (IANA), top-level domain (TLD) operators, and national network information centers (NICs). These organizations maintain accountability mechanisms and support dispute resolution processes, including domain takedowns, mitigation of impersonation, and remediation of malicious behavior. The governance structures surrounding DNS have evolved over decades and are supported by legal, technical, and operational norms that facilitate cross-jurisdictional trust and stability.

DNS also enables policy enforcement at the infrastructure level. For example, country-code top-level domains (ccTLDs) and regionally scoped authoritative servers allow for jurisdictional boundaries to be respected, which may be relevant in agent deployments subject to regulatory constraints. Additionally, DNSSEC and DANE provide cryptographic assurances that bind domain identities to certificates, supporting automated trust decisions without requiring centralized registries or manual validation.

In contrast, alternative discovery proposals— such as blockchain-based registries or neutral third-party systems— lack formal integration with existing governance bodies and may be susceptible to fragmentation, influence, or jurisdictional ambiguity. These models do not currently offer the same level of operational recourse or reliability. Leveraging DNS for agent discovery allows organizations to build on trusted infrastructure while maintaining compatibility with existing governance frameworks, providing a stable foundation for innovation as more experimental models mature.

Establishing a new governance structure based on new agent discovery protocols would be time consuming, involve all stakeholders agreeing on the structure, establishing the structure in a specific

jurisdiction (which may not be suitable for all parties), and involve creating the technical infrastructure to support it. The DNS already has all of these components in place.

2.1.4. Sovereignty

The DNS supports decentralized control while maintaining global interoperability. Organizations may operate their own authoritative DNS infrastructure, publish service records independently, and retain full control over how their services are discovered. This model enables entities to maintain autonomy over their service exposure and operational boundaries without reliance on centralized intermediaries which may not share their same interests.

Proposed alternatives— such as blockchain-based registries, single-purpose generic top-level domains (gTLDs), or third-party discovery networks— often introduce opaque governance models, unclear trust boundaries, and dependencies on external infrastructure. Although these systems may be described as “decentralized,” they frequently concentrate control within protocol maintainers, validator networks, or registry operators, which may not be accountable to the organizations they serve.

DNS provides a proven and extensible registry model that supports secure, independent publication of agent metadata. It aligns with principles of digital sovereignty and operational self-determination, allowing organizations to participate in agent ecosystems without relinquishing control over discovery infrastructure.

2.1.5. Commerce

A critical consideration in the design of agent discovery infrastructure is the potential for commercialization to compromise integrity. In the absence of established norms or constraints, new registry-based discovery mechanisms may evolve into visibility-driven platforms, where agent discoverability is influenced by paid placement rather than technical merit, trust, or reputation. This introduces incentives that prioritize commercial interests over operational reliability and security, undermining the predictability and neutrality of agent interactions and the free and open Internet as a whole.

DNS, by contrast, is built on open standards and governed by transparent, non-commercial processes. It provides mechanisms for validation and authentication, such as DNSSEC, and certificate bindings via DANE that are not subject to monetization. The resolution process is deterministic and not influenced by financial incentives, preserving the integrity of service discovery.

Leveraging DNS ensures that agent discovery remains a technical function grounded in verifiable identity and operational metadata, rather than a commercial battleground. This supports fair access and trustworthy interactions across heterogeneous agent ecosystems.

2.1.6. Application

Regardless of how agent discovery is initiated, agent-to-agent communication ultimately intersects with existing network infrastructure. Agents may require access to internal services, external APIs, or shared data repositories, all of which are typically addressed through conventional networking and resolution mechanisms. Introducing a separate discovery protocol that fragments the location and representation of service metadata introduces architectural complexity and increases the risk of cascading failures across systems.

Operators would be required to manage multiple layers of resolution logic, synchronization mechanisms, and failure domains, which complicates deployment and undermines operational reliability. DNS, by contrast, is already embedded in the fabric of networked applications and provides a natural point of integration for discovery. It enables metadata publication to remain proximate to the services it describes, reducing latency, simplifying architecture, and aligning with existing operational models. This approach supports innovation in agent systems while avoiding brittle dependencies and minimizing architectural drift.

DNS is predictable for agents to use for the purposes of discovery, requiring no advanced configurations to consume the service, rather it is embedded in all pieces of the application stack eliminating the need to retrofit legacy devices to operate in an agent-to-agent environment.

2.1.7. Granularity

DNS supports fine-grained control over service discovery through its hierarchical zone structure and flexible record types. Organizations may define zones and records at varying levels of specificity, including per-host, per-service, per-environment (e.g., staging vs. production), or per geographic region. This granularity enables precise targeting and segmentation of agent capabilities, facilitating operational clarity and policy enforcement.

Service Binding (SVCB) records further enhance granularity by allowing metadata to be associated with individual service endpoints. This includes transport preferences, supported protocols, and application-specific parameters. Combined with DNS Service Discovery

(DNS-SD), these features allow agents to discover services with high specificity, reducing ambiguity and improving interoperability. The DNS model supports modular deployment and incremental adoption, making it well-suited for diverse agent architectures and operational contexts.

2.1.8. Performance

DNS is well-suited to meet the scale and responsiveness requirements of agent discovery, where billions of agents may generate trillions of queries. Its architecture supports high-throughput, low-latency resolution through mechanisms such as caching, retry logic, and autom,atic zone distribution to secondary servers. Recursive resolvers may be further optimized by locally hosting the root zone [RFC8806], reducing lookup latency and improving fault tolerance.

Additional features such as pre-fetching and negative caching allow resolvers to anticipate agent behavior and reduce query overhead. Country code (ccTLD) and Generic Top Level Domain (gTLD) operators have global infrastructure to support localized resolution and improve performance for AI-specific applications. These capabilities enable scalable deployment without requiring protocol redesign or new infrastructure layers.

Furthermore, DNS-based discovery enables immediate connection establishment. Once an agent endpoint is resolved, transport protocols such as QUIC may be used to initiate communication directly, benefiting from features such as 0-RTT and multiplexed streams. This eliminates the need for a separate handshake or negotiation phase for discovery, reducing latency and improving throughput in high-volume environments. DNS thus provides a performant and extensible foundation for agent discovery at global scale.

2.1.9. Lifecycle / Management

DNS provides robust provisioning and lifecycle management capabilities that align with the dynamic nature of agent-based systems. Mechanisms such as Incremental Zone Transfer (IXFR) and Full Zone Transfer (AXFR) allow for rapid propagation of zone updates across authoritative servers. When backed by database-centric DNS implementations, zone data may be treated as replicable state, enabling near-real-time onboarding or decommissioning of agents.

Operational observability is supported through query logging, zone monitoring, and resolver telemetry, which can be used to track agent utilization patterns and identify stale or inactive records.

These lifecycle management capabilities are mature, widely supported, and compatible with existing operational tooling. They enable high-churn agent ecosystems to be managed efficiently without requiring the development of new provisioning, monitoring, or cleanup protocols.

2.1.10. Trust

In agent discovery, trust encompasses more than identity verification — it includes confidence in the provenance, behavior, and reliability of the agents being discovered. The DNS provides a recognizable and auditable namespace rooted in organizational domains (e.g., microsoft.com, openai.com), enabling systems to make informed decisions about agent interactions based on domain ownership and established reputational context. This is particularly critical in AI environments, where agents may initiate autonomous actions or access sensitive data.

DNS enables organizations to publish agent endpoints under domains they control, establishing a clear chain of accountability. This trust model supports cryptographic validation through DNSSEC and DANE, allowing agents to verify domain ownership and certificate bindings without manual intervention. By contrast, discovery mechanisms lacking verifiable lineage introduce significant risk and complexity, particularly when agents are discovered through opaque or ungoverned registries.

The BANDAID model builds on the DNS' s trust infrastructure to support safe and scalable agent discovery. It allows agents to assess the authenticity and authority of remote endpoints prior to interaction, reducing the likelihood of misrouting, impersonation, or unauthorized access. This trust-validation-first approach is foundational to enabling secure automation in agent ecosystems.

2.1.11. Agent Properties

As agent-to-agent protocols evolve, it is likely more metadata will need to be signaled in the discovery phase to allow agents to make more informed decisions. An example of how DNS provides a platform to facilitate this is below.

Service Binding (SVCB) records, as defined in [RFC9460], support extensible key-value parameters that may be registered through the IANA SVCB Parameter Registry. These parameters can provide agent systems with operational context prior to session initiation, enabling informed decisions about whether and how to engage with a remote agent. It is possible to publish metadata such as input (e.g. expected token counts) or cost (e.g. cost per token bundle) directly within DNS records.

This model supports a trust-validation-first approach to agent interaction, where expectations around resource usage and pricing are transparently advertised. It reduces the need for handshake-based negotiation or out-of-band signaling, and allows for deterministic, cacheable discovery workflows. For instance, by embedding cost-related metadata in DNS, discovery becomes not only scalable and performant, but also semantically rich, supporting economically-aware agent ecosystems without introducing new protocols or registries.

2.1.12. Control

Agent discovery mechanisms must operate within the constraints of enterprise security policies, regulatory compliance requirements, and operational boundaries. DNS enables the presentation of distinct DNS views based on the source of the query. This allows organizations to expose different sets of agent records to internal versus external resolvers. For example, internal agents may resolve _a2a._agents.internal.example.com to discover services not intended for public exposure, while external agents receive responses only for _a2a._agents.example.com containing externally authorized endpoints.

Enterprise-grade network controls such as firewalls, Response Policy Zones (RPZs), and access control lists (ACLs) provide additional enforcement mechanisms. These controls may restrict outbound DNS queries to designated resolvers, filter inbound queries to approved zones, and block unauthorized resolution attempts. This ensures that agent discovery adheres to organizational boundaries and complies with internal security policies. For example, an enterprise may configure its network security to allow internal agents to query only *.internal.example.com zones, while blocking access to external agent zones unless explicitly permitted. Similarly, DNS servers may enforce query logging and rate limiting to detect anomalous behavior or potential abuse.

Segmented discovery via DNS aligns with compliance requirements related to data residency, privacy, and access control. By scoping agent visibility to specific zones and enforcing resolution boundaries, organizations can ensure that discovery mechanisms respect jurisdictional constraints, internal governance models, and

support auditability. Query logs, zone access telemetry, and resolver behavior can be monitored to verify that agent interactions conform to policy. This provides a foundation for regulatory reporting, incident response, and continuous compliance validation without the need for extensive additional operations that require more technical staffing or specialized solutions.

2.2. Mechanisms for Successful Agent Communication

The explicit goal of the DNS [RFC1035] is to provide a mechanism for name resolution, enabling agent discovery without requiring new protocols, trust anchors, or infrastructure layers. Leveraging DNS for agent discovery ensures compatibility with existing operational models, supports incremental deployment, and avoids the risks associated with experimental or ungoverned discovery mechanisms.

2.2.1. Service Binding Records, aka SVCB

The SVCB resource record as defined in [RFC9460], enables DNS to convey structured service metadata that facilitates optimized connection establishment. SVCB records support the advertisement of protocol identifiers (via ALPN), transport parameters (e.g., port numbers), endpoint hints (e.g., IP addresses), and extensible key-value attributes. These capabilities are particularly relevant for agent-to-agent communication, where protocol negotiation and metadata exchange must occur without human intervention.

In the context of BANDAID, SVCB records are used to publish agent-specific service endpoints under structured leaf zones (e.g., _chat._agents.example.com). These records allow querying agents to retrieve operational parameters prior to initiating a session, including supported protocols, privacy features (e.g., Encrypted Client Hello via ECH), and failover configurations. This eliminates the need for heuristic-based connection attempts and reduces round-trip latency.

SVCB records also support protocol agility and privacy-preserving features. For example, agents may advertise support for QUIC, HTTP/3, or custom agent-to-agent protocols via ALPN declarations. Operators may specify alternate endpoints or migrate services across domains without relying on CNAME indirection, improving performance and reducing DNS resolution complexity.

When paired with attribute leaf zones and custom SVCB parameters, this mechanism enables fine-grained discovery of agent capabilities. For instance, an agent querying _a2a._agents.example.com may receive an SVCB record indicating supported modalities (e.g., chat, image-to-text), expected input formats, and cost-related metadata (e.g., token pricing). This structured discovery model supports deterministic, cacheable, and semantically rich interactions between agents.

2.2.2. DNS Security Extensions, aka DNSSEC

DNSSEC as defined in [RFC4033], [RFC4034], and [RFC4035], enables resolvers and clients to verify that a response originates from an authoritative source (authenticity) and DNS data has not been modified in transit (data integrity). In the context of BANDAID, DNSSEC serves as a foundational trust mechanism for agent discovery. By signing agent-related resource records— such as SVCB, TXT, or custom metadata— organizations can ensure that querying agents receive authentic and tamper-proof information. This is particularly critical for agent-to-agent communication, where autonomous systems must make trust decisions without human oversight.

DNSSEC supports the establishment of cryptographic trust anchors via Delegation Signer (DS) records and facilitates secure delegation of authority across zone boundaries. This allows organizations to publish agent metadata under subdomains (e.g., _a2a._agents.example.com) while maintaining verifiable control over the zone hierarchy.

Furthermore, DNSSEC eliminates the need for hardcoded trust lists or centralized validation services. Agents can validate responses using the DNSSEC chain of trust, enabling decentralized and scalable trust verification. This model supports zero-trust architectures and aligns with modern security principles by minimizing implicit trust and enforcing explicit validation.

When combined with DANE (DNS-Based Authentication of Named Entities), DNSSEC enables binding of TLS certificates to DNS names, further enhancing the authenticity of agent endpoints. This integration supports secure session initiation and mitigates risks associated with certificate mis-issuance or man-in-the-middle attacks. The use of DANE may improve performance and enable automation in the dynamicism of agent lifecycle.

2.2.3. DNS Service Discovery, aka DNS-SD

DNS-SD as defined in [RFC6763], extends the capabilities of the Domain Name System to support service-type-based discovery. Unlike traditional DNS resolution, which requires prior knowledge of a specific hostname, DNS-SD enables clients to discover services based on their functional type (e.g., _http._tcp, _printer._udp) within a given domain.

In the context of BANDAID, DNS-SD provides a mechanism for AI agents to discover other agents or services based on declared capabilities rather than explicit names. For example, an agent may issue a query for _data-cleaner._a2a._agents.example.com to locate services capable of performing data sanitization tasks. This type-based discovery model supports dynamic ecosystems where agent roles and capabilities may evolve over time.

DNS-SD operates over both multicast DNS (mDNS) and unicast DNS, allowing for flexible deployment in local networks and globally scoped domains. In enterprise or cloud environments, unicast DNS-SD enables structured and policy-compliant discovery across organizational boundaries. When combined with SVCB records, DNS-SD allows agents to retrieve detailed service metadata, including transport preferences, protocol support, and operational parameters.

This model supports federated agent architectures, where multiple agents may advertise similar capabilities under different domains. By leveraging DNS-SD, agents can perform capability-based queries and receive a list of candidate services, each accompanied by structured metadata for evaluation and selection.

DNS-SD also supports extensibility through TXT records and custom service naming conventions, enabling organizations to encode additional attributes relevant to agent interaction (e.g., supported data formats, authentication requirements, or cost models). These features make DNS-SD a suitable foundation for scalable, interoperable, and semantically rich agent discovery workflows.

2.2.4. DNS-Based Authentication of Named Entities, aka DANE

DANE as specified in [RFC6698] and [RFC7671], enables domain owners to associate Transport Layer Security (TLS) certificates with DNS names using TLSA resource records. When protected by DNSSEC, these records provide cryptographically verifiable bindings between domain names and public keys, allowing clients to authenticate services without relying solely on external certificate authorities (CAs).

In the context of BANDAID, DANE enhances the trust model for agent-to-agent communication by enabling agents to validate the authenticity of remote endpoints based on domain-controlled assertions. This is particularly relevant in automated environments where agents must establish secure sessions without manual certificate validation or user interaction.

By publishing TLSA records under agent-specific service names (e.g., _a2a._tcp.agent.example.com), organizations can declare the expected certificate fingerprint or public key for a given agent endpoint. Querying agents can then validate the TLS connection against the DANE record, ensuring that the certificate presented during session initiation matches the domain's published intent.

This mechanism mitigates risks associated with certificate mis-issuance, man-in-the-middle attacks, and reliance on third-party trust anchors. It also supports deterministic validation workflows, which are critical for agents operating in headless or GUI-less environments where interactive certificate warnings cannot be resolved.

When combined with DNSSEC, DANE provides a robust, decentralized, and verifiable trust infrastructure for agent discovery and communication. It aligns with zero-trust principles by enabling domain owners to assert and control trust relationships directly through DNS, without introducing additional registries or validation services.

2.2.5. Domain Control Validation, aka DCV

DCV refers to the process by which an entity demonstrates authoritative control over a DNS domain. As described in draft-ietf-dnsop-domain-verification-techniques, DNS-based DCV typically involves the placement of a DNS record—most commonly a TXT record—containing a challenge token or assertion at a designated location within the domain. This record is then queried by an Application Service Provider (ASP) to confirm control.

In the context of BANDAID, DCV provides a mechanism for agents to assert delegated authority on behalf of a domain. For example, an organization may publish a TXT record under a structured leaf zone (e.g., _agent-roles._a2a._agents.example.com) containing metadata such as:

```
ai-role=data-cleaner; crm=salesforce; access=readonly
```

This record signals that a specific agent is authorized to perform scoped operations under the domain's authority. When protected by DNSSEC, such assertions become cryptographically verifiable, mitigating risks of spoofing or unauthorized delegation.

DCV within BANDAID supports both ephemeral and persistent validation models. Ephemeral validation may be used for short-lived agent credentials or session-based delegation, while persistent records enable long-term authorization signaling. TTL and expiration considerations, as discussed in the draft, are critical to ensuring that validation records do not persist beyond their intended scope.

Furthermore, BANDAID encourages the use of application-specific naming conventions and token formats to avoid service confusion and collision. Validation records should be scoped to the intended service and include unguessable tokens or structured metadata to prevent unauthorized reuse across federated agent ecosystems.

2.2.6. Service Resilience

The DNS architecture is inherently resilient due to its distributed, hierarchical design. Authoritative zones are replicated across multiple servers, and recursive resolvers implement caching, retry logic, and failover mechanisms to ensure continuity of resolution. Failover, anycast, and replication continue to add resiliency and redundancy provided through the DNS.

This fault-tolerant model supports agent discovery even in the presence of partial network outages, denial-of-service conditions, or upstream failures. Unlike centralized registries or proprietary APIs, DNS does not rely on a single point of availability, making it suitable for high-availability agent ecosystems operating across diverse network environments.

2.2.7. Observability, Auditability

DNS provides built-in observability through query logging, resolver telemetry, and zone monitoring. These features enable operators to track agent discovery patterns, detect anomalous behavior, and perform forensic analysis of agent interactions. This auditability is critical in regulated environments where agent actions must be attributable and verifiable, and where discovery mechanisms must align with enterprise governance models.

Logs may be used to validate compliance with access policies, identify stale or misconfigured records, support incident response workflows, and/or be used in lifecycle management of unused or oversubscribed agents.

2.2.8. Interoperability

BANDAID is designed to be agnostic to agent implementation frameworks. Whether agents are built using open-source libraries, proprietary orchestration platforms, or custom runtimes, DNS-based discovery provides a common substrate for metadata publication and retrieval. This interoperability ensures that agents operating under different protocols, vendors, or administrative domains can participate in shared discovery workflows without requiring protocol translation or framework-specific integration. DNS's ubiquity across operating systems and network stacks further reinforces its suitability as a universal discovery layer.

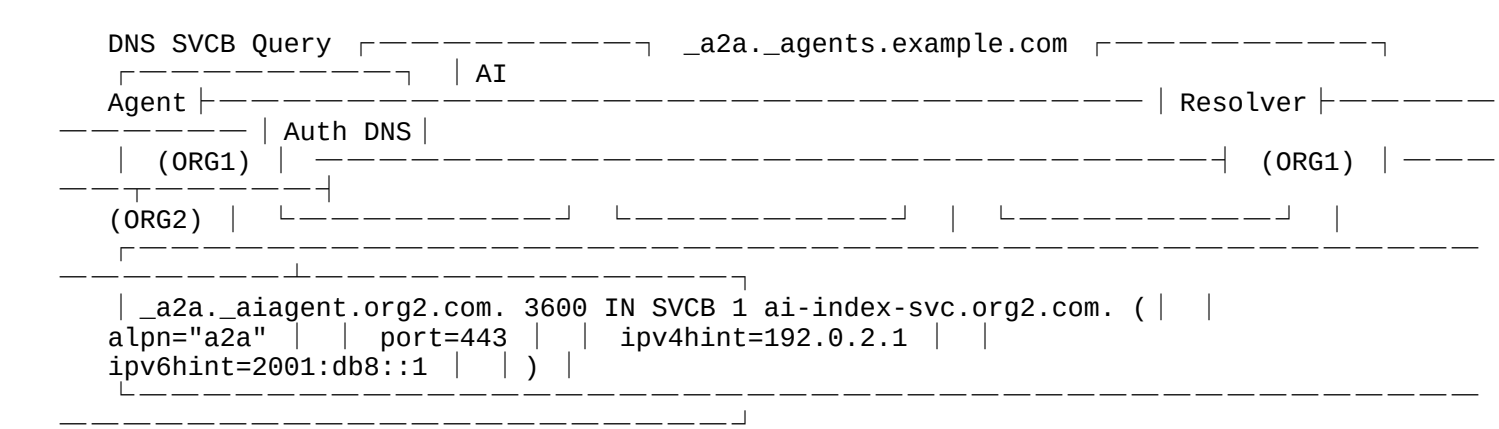
2.2.9. Multitenancy, Federation

DNS's hierarchical namespace supports multi-tenant and federated discovery models. Organizations may delegate subdomains to tenants (e.g., customer1._agents.vendor.com) or publish agent metadata under federated zones (e.g., region1._a2a.example.org). This enables scoped discovery, policy enforcement, and operational isolation across tenants, departments, or geographic regions. Combined with DNSSEC and access controls, this model supports secure delegation and trust signaling in complex agent ecosystems, including SaaS platforms and cross-organizational collaborations.

2.3. Architecture

2.3.1. Example Communication

Agent (org1) wants to find other agents provided by another entity (org2) and discover/verify capabilities



2.3.2. Example Record

```
_a2a._aiagent.org2.com. 3600 IN SVCB 1 ai-index-svc.org2.com. (
alpn="a2a" port=443 ipv4hint=192.0.2.1 ipv6hint=2001:db8::1 )
```

_a2a._aiagent.example.com.: The service name, following the SVCB naming convention (_service._agents.example.com). 3600: TTL (Time to Live) in seconds. IN SVCB: The record type. 1: SVCB priority. 0 is for alias mode; 1+ is for service mode. ai-index-svc.org2.com.: The target name of the service. alpn="a2a": Specifies the ALPN protocol identifier. This is where your custom protocol is declared. port=443: The port on which the service is available. ipv4hint and ipv6hint: Optional hints for clients to connect directly.

2.3.3. Example Use Cases

A user instructs their internal agent to “clean up Salesforce contacts based on this email.” The agent must discover Salesforce’s authorized agents, validate its own delegation to act on behalf of the enterprise, and initiate a secure session. BANDAID enables this by publishing agent endpoints and roles under DNS zones controlled by each organization.

A research consortium deploys agents across multiple institutions. Each institution publishes its agents under its own domain (e.g., _a2a._agents.universityA.edu), allowing collaborators to discover services based on capability (e.g., _data-annotator._a2a.universityB.edu) while respecting institutional boundaries and trust models.

A SaaS provider hosts agents for multiple customers. Each customer’s agents are published under tenant-specific zones (e.g., customer1._agents.saas.com), enabling scoped discovery and policy enforcement. BANDAID supports this model through hierarchical zone delegation and metadata-rich SVCB records.

Lightweight agents deployed on mobile or edge devices require low-latency, cacheable discovery mechanisms. DNS’s distributed architecture and support for SVCB hints (e.g., IP addresses, preferred protocols) enable efficient resolution and connection bootstrapping in constrained environments.

In regulated industries, agents must operate within jurisdictional boundaries and maintain audit trails of interactions. DNS supports geographic scoping via ccTLDs and split-horizon configurations, while query logging and DNSSEC provide observability and integrity guarantees.

3. BANDAID Overview

The BANDAID model is designed for incremental and non-disruptive deployment within existing DNS infrastructure. It introduces no new DNS message formats, opcodes, response codes, or resource record types. Instead, it defines a structured namespace convention and usage profile for existing record types... primarily SVCB, TXT, and TLSA all within designated leaf zones (e.g., _a2a._agents.example.com).

Organizations may adopt BANDAID by publishing agent metadata under delegated subdomains, leveraging DNSSEC for integrity and authenticity, and optionally implementing Domain Control Validation (DCV) to signal delegated authority. These zones may be exposed selectively via split-horizon DNS, enabling differentiated discovery views for internal and external agents. No changes are required to recursive or authoritative DNS server implementations beyond standard support for DNSSEC and SVCB.

This model supports opt-in adoption, allowing agent operators to publish discovery metadata without coordination with external registries or protocol maintainers. It is compatible with existing DNS tooling, including zone provisioning systems, monitoring platforms, and resolver configurations. BANDAID is therefore suitable for deployment across enterprise, cloud, and federated environments, and may coexist with other discovery mechanisms without conflict.

3.1. Zone and Other Requirements

3.1.1. Delegation and Chain of Trust

An external authoritative zone used for the purposes of BANDAID MUST use DNSSEC [RFC4033]. The BANDAID external zone MUST establish a complete chain of trust to a publicly recognized trust anchor. For zones delegated from a public parent, DS records MUST be present and maintained at the parent zone. DS digests MUST use SHA-256 (Digest Type 2) as specified in [RFC4509]; SHA-1 (Digest Type 1) MUST NOT be used. Where feasible, automation of DS maintenance SHOULD be performed using CDS and CDNSKEY as specified in [RFC7344] and operationalized per [RFC8078].

3.1.2. Algorithms and Parameters

Authoritative signers for the BANDAID External Zone MUST use DNSSEC algorithms and key sizes consistent with current implementation and usage guidance in [RFC8624]. Implementations SHOULD prefer elliptic-curve or EdDSA algorithms for efficiency and security, specifically ECDSA P-256 (Algorithm 13; [RFC6605]) or Ed25519 (Algorithm 15; [RFC8080]). Zones MUST NOT rely on RSA/SHA-1 signatures. If RSA is used for transitional reasons, key sizes SHOULD be 2048 bits and migration to ECDSA or EdDSA SHOULD be planned.

Separation of roles between a Key-Signing Key (KSK) and a Zone-Signing Key (ZSK) is RECOMMENDED for operational agility, consistent with practices in [RFC6781]. Private keys SHOULD be protected by appropriate controls (e.g., HSMs or equivalent) commensurate with organizational risk.

3.1.3. Authenticated Denial of Existence

The BANDAID External Zone MUST provide authenticated denial of existence. NSEC3 [RFC5155] is RECOMMENDED to mitigate trivial zone enumeration for zones where record names or labels might reveal sensitive information. If NSEC3 is used, the number of iterations SHOULD be kept modest to balance CPU cost for signers and validators, per guidance in [RFC6781]. Use of NSEC with White Lies is OPTIONAL where operationally justified.

3.1.4. Signature Validity

Signature validity windows MUST be chosen to balance cache efficiency and replay risk. In general, a signature lifetime on the order of days to a small number of weeks is RECOMMENDED, with staggered resigning to avoid mass expiration events [RFC6781]. Resource Record TTLs MUST be set to values that support timely updates to BANDAID capability data while allowing effective caching; operational defaults SHOULD be validated against change frequency. Negative caching behavior MUST follow [RFC2308].

3.1.5. Key Rollover

Zones MUST support planned key rollovers and emergency key replacement consistent with the procedures in [RFC6781]. Pre-publication and double-signature techniques SHOULD be used to minimize validation failures during ZSK and KSK rollovers. Parent DS changes SHOULD be orchestrated using CDS/CDNSKEY signaling [RFC7344][RFC8078]. Post-compromise recovery procedures MUST be documented, including revocation, re-signing, and DS replacement

timelines.

3.1.6. Transport and Message Size

Because DNSSEC increases response sizes, authoritative servers for the BANDAID External Zone MUST implement EDNS(0) per [RFC6891] and MUST support TCP fallback for queries per [RFC7766]. Authoritative operators SHOULD configure UDP response sizing to minimize IP fragmentation risk and SHOULD ensure that large responses (e.g., DNSKEY, NSEC3 chains) remain retrievable via TCP.

3.1.7. Transfer and Integrity

When secondary authoritative servers are used, zone transfers MUST be authenticated and protected with TSIG [RFC2845]. Use of DNS NOTIFY [RFC1996] and incremental transfer (IXFR; [RFC5936]) is RECOMMENDED to reduce propagation latency and bandwidth. To provide at-rest integrity verification between primaries and secondaries, publishing a ZONEMD record is RECOMMENDED where supported [RFC8976].

3.1.8. Monitoring

Operators MUST monitor DNSSEC validity from multiple vantage points to detect stale signatures, DS/DNSKEY mismatches, and other validation failures. Alarms SHOULD be configured for impending signature expiration, key compromise indicators, and DS inconsistencies. Signing infrastructure SHOULD be tested under load and failure scenarios consistent with the scalability requirements of this specification.

3.1.9. Abuse Resistance

Authoritative servers for the BANDAID External Zone SHOULD implement DNS Cookies [RFC7873][RFC9018] to reduce off-path spoofing and amplification risks. Operators SHOULD follow best practices to limit amplification and to shape responses under attack, consistent with maintaining DNSSEC validity.

3.1.10. BANDAID Records

All BANDAID-specific discovery records (e.g., SRV [RFC2782], SVCB/HTTPS [RFC9460], TXT/URI [RFC7553] used for capability descriptors, and any BANDAID-defined RRTypes) MUST be signed and published in the BANDAID External Zone. Where BANDAID endpoints rely on TLS, publication of DANE TLSA records SHOULD be used to bind endpoint certificates to DNSSEC-validated names [RFC6698][RFC7671]. Resolver behavior consuming BANDAID data MUST treat DNSSEC-bogus responses as failures and MUST NOT act on unsigned or invalidly signed discovery

data.

3.1.11. Performance Optimization(s)

The discovery namespace for BANDAID SHOULD be structured to minimize query depth, RRset size, and update blast radius. To that end, implementers SHOULD allocate per-service, per-agent leaf names and avoid aggregating high-cardinality data under a single owner name. A deterministic mapping (e.g., a stable hash of an Agent Identifier) to leaf labels SHOULD be used to support sharding and parallel administration. Where cardinality requires, sub-zones MAY be delegated (zone cuts) to distribute authority and update load.

Each agent service endpoint SHOULD be published using SVCB in ServiceMode (or HTTPS RR for HTTPS endpoints) to convey connection parameters and capability locators with a single lookup [RFC9460]. SVCB “address hints” (ipv4hint, ipv6hint) SHOULD be used to reduce A/AAAA follow-up queries; resolvers MAY still validate final addresses via canonical resolution. Where human-friendly names are required, SVCB AliasMode (priority 0) MAY be used to map from a stable alias to a hashed leaf, avoiding record duplication while preserving cache locality.

Authoritative servers MUST support EDNS(0) [RFC6891] and TCP fallback [RFC7766]. Operators SHOULD target response sizes that avoid IP fragmentation on common paths (e.g., 1232 bytes on IPv6), preferring compression and layered indirection over oversized RRsets. TTLs MUST be chosen to reflect the volatility of capability data; index/indirection records may use longer TTLs than frequently changing endpoint or capability descriptors. Negative caching MUST conform to [RFC2308]. When high update rates are expected, IXFR and NOTIFY SHOULD be used to reduce propagation latency.

Resolvers and authoritative operators SHOULD take advantage of SVCB’s ALPN, port, and (for HTTPS) ECH parameters to enable connection pre-establishment and reduce round-trips, subject to client policy and security posture [RFC9460]. Where privacy policies prohibit client geolocation, EDNS Client Subnet SHOULD NOT be relied upon for performance.

A representative naming pattern is shown below; the exact label order is deployment-specific, but the leaf per service, per agent rule applies:

```
‘’’ ; Hashed, per-agent, per-service leaf
a4k2f9._mcp._bandaid.example.org.  600 IN SVCB 1 svc-
a4k2f9.example.org.  alpn="h2,h3" port=443 ipv6hint=2001:db8::5
ipv4hint=192.0.2.5
```

```
; Optional alias for a friendlier owner name
billing._mcp._bandaid.example.org. 300 IN SVCB 0
a4k2f9._mcp._bandaid.example.org. '''
```

3.1.12. Customization(s)

BANDAID deployments MAY define additional SVCB parameters (SvcParamKeys) to convey agent-specific capability metadata, provided that interoperability safeguards in [RFC9460] are observed. During experimentation, unregistered keys MUST use the numeric keyNNNNN presentation form, and any client behavior that depends on them MUST be gated via the mandatory SvcParam to ensure downgrade safety. This specification defines the following provisional SvcParamKeys for BANDAID (names are illustrative; production deployments MUST register through IANA per [RFC9460] or use keyNNNNN until standardized):

cap — a capability descriptor locator or inline identifier (e.g., a URN or compact JSON-Ref) that identifies the agent' s advertised capability schema/version. cap-sha256 — a base64url-encoded SHA-256 digest of the canonical capability descriptor to support integrity checks and cache revalidation. bap — a comma-separated list of BANDAID Application Protocols and versions understood by the endpoint (e.g., bap="a2a/1,mcp/1,acp/2"). This is distinct from transport-level alpn. policy — a URI or URN identifying a policy bundle applicable to this agent (e.g., jurisdiction, data handling class) for client-side selection. realm — an opaque token for multi-tenant scoping or authz realm selection during protocol bootstrapping.

Clients that require any of these parameters MUST verify their presence via the mandatory key; otherwise, the SVCB record MUST be ignored. Example:

Single-RRTYPE publication with custom params (experimental keys shown) a4k2f9._mcp._bandaid.example.org. 600 IN SVCB 1 svc-a4k2f9.example.org. alpn="h2" port=443 ipv4hint=192.0.2.5 mandatory=alpn,port,key65001,key65002,key65010 key65001="cap=urn:cap:example:mcp:invoice.v1" key65002="cap-sha256=yvZ0n7q8bE2gYkz8m1j1s0yQG0mC2F6qj3b9pVb6Gk0" key65010="bap=a2a/1,mcp/1" Where endpoints are HTTPS, the HTTPS RR variant SHOULD be used to co-locate transport parameters such as ech with capability metadata; otherwise, generic SVCB MAY be used. Future standardization SHOULD define the exact syntax (e.g., ABNF) and registry policy for these keys, including error handling for malformed or conflicting parameters. Until registration is complete, deployments MUST treat unknown keyNNNNN parameters as opaque and MUST NOT infer semantics without out-of-band agreement.

4. Implementation Guidance

4.1. AI Operators

4.1.1. Separation of Roles

Operators MUST separate authoritative servers for BANDAID zones from recursive resolvers to prevent cache poisoning and reduce operational coupling. Authoritative servers MUST NOT perform recursion. Recursive resolvers SHOULD be deployed close to query sources (e.g., within organizational networks or edge PoPs) to minimize latency and reduce upstream load.

4.1.2. Local Root Zone Copy

Recursive resolvers serving BANDAID traffic SHOULD maintain a local copy of the root zone as specified in [RFC8806]. This eliminates dependency on external root servers for priming queries, reduces resolution latency, and improves resilience during partial Internet outages or denial-of-service events targeting root infrastructure.

4.1.3. Aggressive Caching and Prefetch

Resolvers MUST implement RFC-compliant caching and SHOULD enable the aggressive use of DNSSEC-validated cache to synthesize negative answers from NSEC/NSEC3 and reduce upstream lookups, consistent with [RFC8198]. This behavior improves latency, lowers authoritative load, and can mitigate certain attack patterns by answering within validated non-existence ranges.

Resolvers SHOULD implement serve-stale as specified in [RFC8767]. When authoritative servers are unreachable or fail to refresh data, a resolver MAY return expired records from its cache to preserve availability, subject to policy caps on staleness. [RFC8767] updates the TTL semantics to allow use beyond expiration under these exceptional conditions and RECOMMENDS capping staleness on the order of days, with a typical cap of 7 days. When serving stale data, the TTL in the response MUST be set to a value greater than zero (with 30 seconds RECOMMENDED) so clients and intermediaries do not cache a zero-TTL response indefinitely.

Resolvers implementing serve-stale SHOULD follow the timer guidance in [RFC8767]: continue background refresh (“stale-while-revalidate”), bound total resolution work with a query-resolution timer, avoid excessive re-tries with a failure-recheck timer (no more frequently than ~30 seconds is RECOMMENDED), and ensure client responsiveness with a client-response timer (on the order of ~1.8 seconds is RECOMMENDED). These controls balance resiliency with freshness and protect upstream authorities during outages or attack conditions.

Operators MUST consider DNSSEC interactions when serving stale data. Stale answers can increase the likelihood that signatures are outside their validity windows; implementations MUST NOT misrepresent validation status (e.g., AD bit handling MUST follow DNSSEC semantics where the bit is only set if all relevant RRsets are cryptographically verified according to local policy).

Prefetch of popular BANDAID names SHOULD be used to keep caches warm, and negative caching MUST follow [RFC2308] to avoid unnecessary upstream traffic during flaps. Where serve-stale is enabled, operators SHOULD monitor stale-answer rates and staleness distribution to ensure that policy caps and refresh behavior meet availability and freshness objectives.

4.1.4. EDNS(0) Transport Resilience

All servers MUST support EDNS(0) [RFC6891] and TCP fallback [RFC7766] to accommodate DNSSEC and SVCB responses that exceed traditional UDP size limits. Operators SHOULD configure UDP response sizing to avoid IP fragmentation (e.g., 1232 bytes for IPv6 paths) and SHOULD monitor truncation rates to detect misconfigurations or path MTU issues.\

4.1.5. Load Distribution

Authoritative servers for BANDAID zones SHOULD be deployed using IP anycast to provide global reachability, load balancing, and DDoS resilience. Recursive resolvers SHOULD be deployed in a geographically distributed manner to minimize latency for agent queries and to provide failover during regional outages.

4.1.6. Monitoring and Telemetry

Operators MUST implement continuous monitoring of query latency, cache hit ratios, DNSSEC validation success rates, and transport fallback events. Alarms SHOULD be configured for anomalies such as sudden TTL exhaustion, signature expiration, or upstream unreachability. Logging MUST respect privacy and regulatory constraints while providing sufficient detail for operational

troubleshooting and compliance audits.

4.1.7. Encrypted Resolver Steering (Informative)

AI clients and operators MAY implement Encrypted DNS Server Redirection (EDSR) to permit an encrypted resolver to redirect a client to another encrypted resolver for performance, locality, or policy reasons Any acceptance of redirection MUST follow the verified-discovery model of DDR [RFC9462], including validation of the designated relationship and TLS authentication of the target resolver. If redirection validation fails or exceeds a single hop, clients MUST continue to use the original resolver configuration. When available, network-designated resolvers provisioned via DNR [RFC9463] take precedence over unsolicited redirection unless local policy dictates otherwise. EDSR is currently an Internet-Draft and is referenced as an OPTIONAL extension.

4.2. AI Consumers

4.2.1. DNS-Based Service Discovery

DNS-Based Service Discovery AI clients that consume BANDAID discovery data MUST implement DNS-based service discovery consistent with [RFC6763] and [RFC9460]. Clients SHOULD query for SVCB or HTTPS records at well-defined service labels (e.g., _mcp._bandaid.example.org.) and interpret alpn, port, and custom SvcParams as specified in this document. Where multiple priorities are returned, clients MUST honor the SVCB priority and weight semantics for selection and failover. Clients MUST validate DNSSEC signatures for all discovery responses and MUST NOT act on unsigned or bogus data.

Clients SHOULD implement caching consistent with TTLs and negative caching rules in [RFC2308]. Prefetching of high-traffic names MAY be used to reduce latency, provided that it does not violate policy or privacy constraints. Clients MUST support EDNS(0) and TCP fallback to handle large responses.

4.2.2. Communication Failure / Retry

When a discovered agent endpoint is unreachable, clients MUST implement retry logic that respects exponential backoff and avoids synchronized retry storms. Clients SHOULD attempt alternate SVCB targets in priority order before declaring failure. Where serve-stale responses are received, clients MUST treat them as advisory and MUST NOT assume freshness beyond the TTL indicated in the response. Clients SHOULD log stale usage for observability and compliance.

4.2.3. Domain Control Validation

To prevent impersonation and unauthorized data access, BANDAID deployments MUST support Domain Control Validation (DCV) using DNS. This mechanism allows application owners to verify that a requesting service is authorized to act on behalf of a domain (e.g., “AI, sync all customer contacts from my approved CRM for acme.org”). DCV MUST follow a challenge-response model where the domain owner publishes a cryptographically strong token in a DNS TXT record under a well-defined label (e.g., _bandaid-challenge.acme.org.). The token MUST be bound to the requesting service identity and have a bounded validity period. Clients performing DCV MUST validate the token against the expected value and expiration before granting delegated access. Tokens SHOULD be single-use and MUST be removed from DNS after successful validation. All DCV interactions MUST occur over DNSSEC-validated channels to prevent spoofing or downgrade attacks.

4.3. AI Developers

4.3.1. Publishing Schema

Publishers SHOULD expose per-service discovery records using SVCB (or HTTPS for HTTP origins) at stable, well-scoped owner names (e.g., agent-id._mcp._bandaid.example.org.). SVCB ServiceMode conveys connection parameters and capability locators in a single round trip; AliasMode MAY be used to map a friendly name to a hashed leaf for sharding without duplicating RRSets. Initial connection parameter keys (alpn, port, address hints) and the mandatory key MUST be honored by clients. [RFC9460]

Minimal example:

```
; ServiceMode SVCB for an MCP-capable agent
a4k2f9._mcp._bandaid.example.org.  600 IN SVCB 1 svc-
a4k2f9.example.net.  alpn="h2,h3" port=443 ipv6hint=2001:db8::5
ipv4hint=192.0.2.5 mandatory=alpn,port
```

SVCB/HTTPS semantics and SvcParam processing MUST follow [RFC9460] to ensure interop and correct fallback.

4.3.2. TTLs, Update Agility

Publishers SHOULD assign longer TTLs to relatively static indirection records (aliases, stable service labels) and shorter TTLs to volatile endpoint/capability records to balance cache efficiency with rollout safety. Consumers will apply negative caching per [RFC2308], so publishers SHOULD avoid unnecessary NXDOMAIN flaps during deployments (e.g., prefer blue/green leaves over in-place deletions).

4.3.3. Capability Retrieval

When capability metadata is referenced via URI from SVCB/HTTPS, retrieval MUST occur over authenticated, confidential channels. Publishers SHOULD offer TLS 1.3 and MAY bind endpoint identity via DANE TLSA where resolvers validate DNSSEC; otherwise, WebPKI validation applies. Operational guidance for DANE usage (certificate usage selections, rollover) SHOULD follow [RFC7671] [RFC8446] [RFC6698]

Where using HTTPS RRs, publishers MAY advertise the ech SvcParam to enable Encrypted Client Hello for privacy; client/server behavior and key distribution MUST follow the semantics described in [RFC9460] for the ech parameter.

4.3.4. Example DCV Flow

Example DCV Flow (Normative Sketch)

* The relying service issues a DCV challenge with a nonce bound to its identity and an expiration time.

* The domain owner publishes the nonce at _bandaid-challenge.domain. as a TXT RR until validation completes.

* The verifier performs a DNSSEC-validated TXT query and compares the presented nonce and binding; on success, access is granted and the TXT is removed.

need a sketch for how this looks

4.3.5. Example Zonefile

‘‘‘ $ORIGIN example.org. $TTL 3600

; -- --- ; SOA / NS ; -- ------------------ @ IN SOA ns1.example.org. hostmaster.example.org. (2025091901 ; serial (YYYYMMDDnn) 7200 ; refresh 1800 ; retry 1209600 ; expire 3600) ; minimum IN NS ns1.example.org. IN NS ns2.example.org.

ns1 IN A 192.0.2.53 ns1 IN AAAA 2001:db8::53 ns2 IN A 192.0.2.54 ns2 IN AAAA 2001:db8::54

; -- --- ; BANDAID discovery (MCP example): per-service, per-agent leaf ; - Leaf is a stable mapping from AgentID -> label (e.g. hashed). ; -

```
   Use SVCB ServiceMode (priority > 0) to bind connection parameters. ;
   ------------------------------------------------------------------
   -

   ; AliasMode to keep a friendly name stable even if the leaf changes
   billing._mcp._bandaid 300 IN SVCB 0 a4k2f9._mcp._bandaid.example.org.

   ; Leaf per agent/service (ServiceMode).  Shorter TTL for agility.
   a4k2f9._mcp._bandaid 600 IN SVCB 1 svc-a4k2f9.example.net. \
   alpn="h2,h3" \ port=443 \ ipv4hint=192.0.2.5 \ ipv6hint=2001:db8::5 \
   mandatory=alpn,port

   ; (Optional) Another service face for the same agent (A2A)
   a4k2f9._a2a._bandaid 600 IN SVCB 1 svc-a4k2f9.example.net. \
   alpn="h2" port=8443 \ mandatory=alpn,port

   ; (Optional) HTTPS RR at apex for web-style consumers of BANDAID
   metadata @ 900 IN HTTPS 1 web-gw.example.net. \ alpn="h2,h3" port=443

   ; ----------------------------------------------------------------
   --- ; Capability / policy signaling via SVCB custom keys
   (experimental) ; Use numeric keyNNNNN until you register IANA
   SvcParamKeys. ; Gate client requirements with "mandatory". ; --------
   ------------------------------------------------------------ ;
   Example: reference a capability descriptor and advertise supported
   app protocols. ; NOTE: Values and key numbers are illustrative.

   a4k2f9._mcp._bandaid 600 IN SVCB 1 svc-a4k2f9.example.net. \
   alpn="h2,h3" port=443 \ mandatory=alpn,port,key65001,key65010 \
   key65001="cap=urn:cap:example:mcp:invoice.v1" \
   key65010="bap=a2a/1,mcp/1"

   ; ----------------------------------------------------------------
   --- ; Domain Control Validation (DCV) for BANDAID authorization ;
   Publish a short-lived token to prove control over example.org. ;
   Remove after successful validation.  Resolver MUST DNSSEC-validate. ;
   ----------------------------------------------------------------
   - _bandaid-challenge 300 IN TXT "bnd-req=svc:crm-
   sync@vendor.example;nonce=3Qz6l8pA;exp=2025-09-19T06:00:00Z"

   ; ----------------------------------------------------------------
   --- ; Optional DANE TLSA for the service endpoint used above. ;
   Owner: _<port>._tcp.<endpoint-FQDN> ; Usage 3 (DANE-EE), Selector 1
   (SPKI), Match 1 (SHA-256) ; Replace the hash with the actual SPKI
   hash of the leaf certificate. ; ------------------------------------
   ------------------------------- _443._tcp.svc-a4k2f9.example.net.
   1800 IN TLSA 3 1 1 (
   0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789AB )
```

```
; ------------------------------------------------------------------
--- ; (Optional) Address records for the service endpoint, if hosted
in-zone ; If hosted out-of-zone (example.net), these will live there
instead. ; ------------------------------------------------------------
------------ svc-a4k2f9 900 IN A 192.0.2.5 svc-a4k2f9 900 IN AAAA
2001:db8::5 web-gw 900 IN A 192.0.2.80 web-gw 900 IN AAAA
2001:db8::80 '''
```

4.3.5.1. How to use this zonefile

* Adjust TTLs for agility vs. cache efficiency. Use longer TTLs on indirection (aliases), shorter TTLs on volatile leaves and DCV. This aligns with DNS caching behavior and negative caching rules [RFC2308].

* Follow SVCB/HTTPS semantics. Clients must honor SvcPriority, AliasMode (priority 0), and ServiceMode parameters, including mandatory, alpn, port, and address hints per [RFC9460].

* DCV token flow. Issue a time-bounded token at _bandaid-challenge.domain and require DNSSEC-validated retrieval (pattern analogous to ACME's DNS-01 in [RFC8555]). Remove on success.

* Bind transport to DNS with DANE (optional). If your relying parties validate DNSSEC, publish TLSA to bind the endpoint's cert/key, using the operational guidance in [RFC7671] (TLSA record format in [RFC6698]).

* Sign the zone. This file is pre-signing. Sign with DNSSEC (DNSKEY/DS/RRSIG, etc.) before deployment; validators must treat unsigned/bogus discovery data as a failure (per your earlier spec). See DNSSEC core specs [RFC4033], [RFC4034], [RFC4035] and operational guidance.

4.3.5.2. Why these records

* SVCB/HTTPS concentrate connection metadata and allow aliasing at apex and service labels, reducing round trips and enabling policy-gated params via mandatory.

* Per-service, per-agent leaves avoid oversized RRsets and allow parallel administration/sharding while keeping alias names stable. This follows performance guidance in SVCB and your BANDAID perf section.

* DCV via TXT mirrors a well-understood, automated control-proof pattern [RFC8555] that fits BANDAID authorization workflows.

* DANE TLSA optionally removes reliance on external PKI anchors for endpoint auth where DNSSEC is deployed [RFC6698], with deployment rules in [RFC7671].

5. Conclusion

The discovery of AI agents at Internet scale introduces requirements for autonomy, security, resilience, and interoperability that cannot be met by ad-hoc or centralized mechanisms. Organizations must advertise capabilities in a manner that is globally unique, cryptographically verifiable, and operationally scalable, while preserving compliance with regulatory and sovereignty constraints. The solution space must account for dynamic lifecycles, selective disclosure, and the ability to integrate with existing infrastructure without introducing new trust anchors or governance choke points.

This document defines BANDAID as a DNS-based discovery framework that leverages the global DNS namespace, DNSSEC for integrity and authenticity, and SVCB/HTTPS records for structured capability advertisement. By building on widely deployed protocols and operational practices, BANDAID provides a predictable entry point for agent discovery, supports hierarchical delegation for organizational autonomy, and enables secure, low-latency resolution at scale. Extensions such as custom SvcParams, DNS-based domain control validation, and optional DANE bindings further strengthen trust and interoperability.

Future work includes standardizing capability schemas, formalizing SvcParam registries for BANDAID metadata, and defining privacy-preserving query mechanisms (like authoritative DOT/H/Q). By aligning with existing Internet standards and operational best practices, BANDAID offers a path to interoperable, secure, and resilient AI agent discovery without reinventing the foundational layers of the Internet.

5.1. Future Work & Unaddressed Portions

Index versus agents publishing their updates, etc. Unresolved in the use case of it being better to have an index which is a list of all agents an org owns, or allow records within the zone to be the record of truth.

For example, a company may only have a few external agents available for use, and if the IETF standardizes ’types’ of AI agents, then perhaps _chat._ai.domain.com or _img2txt._ai.domain.com etc are all different SVCB records.

6. Security Considerations

To add: Lookalikes, takedowns, misbehaving agents, compliance, privacy

7. IANA Considerations

IANA are to consider registering an underscored attribute leaf as part of [RFC8552] for BANDAID.

IANA are to consider designating custom key-value pairs for SVCB parameters to facilitate agent-to-agent application discovery, potentially for cost optimization (token input counts, costs per token bundle, etc.)

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>.

[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>.

8.2. Informative References

[I-D.draft-mozley-aidiscovery] Mozley, J., Williams, N., Sarikaya, B., and R. Schott, "AI Agent Discovery (AID) Problem Statement", Work in Progress, Internet-Draft, draft-mozley-aidiscovery-00, 15 October 2025, <https://datatracker.ietf.org/doc/html/draft-mozley-aidiscovery-00>.

[I-D.draft-rosenberg-ai-protocols] Rosenberg, J. and C. F. Jennings, "Framework, Use Cases and Requirements for AI Agent Protocols", Work in Progress, Internet-Draft, draft-rosenberg-ai-protocols-00, 5 May 2025, <https://datatracker.ietf.org/doc/html/draft-rosenberg-ai-protocols-00>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/rfc/rfc1035>.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, DOI 10.17487/RFC1996, August 1996, <https://www.rfc-editor.org/rfc/rfc1996>.

[RFC2308] Andrews, M., "Negative Caching of DNS Queries (DNS NCACHE)", RFC 2308, DOI 10.17487/RFC2308, March 1998, <https://www.rfc-editor.org/rfc/rfc2308>.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <https://www.rfc-editor.org/rfc/rfc2782>.

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <https://www.rfc-editor.org/rfc/rfc2845>.

[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>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/rfc/rfc4034>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <https://www.rfc-editor.org/rfc/rfc4035>.

[RFC4509] Hardaker, W., "Use of SHA-256 in DNSSEC Delegation Signer (DS) Resource Records (RRs)", RFC 4509, DOI 10.17487/RFC4509, May 2006, <https://www.rfc-editor.org/rfc/rfc4509>.

[RFC5155] Laurie, B., Sisson, G., Arends, R., and D. Blacka, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", RFC 5155, DOI 10.17487/RFC5155, March 2008, <https://www.rfc-editor.org/rfc/rfc5155>.

[RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <https://www.rfc-editor.org/rfc/rfc5936>.

[RFC6605] Hoffman, P. and W.C.A. Wijngaards, "Elliptic Curve Digital Signature Algorithm (DSA) for DNSSEC", RFC 6605, DOI 10.17487/RFC6605, April 2012, <https://www.rfc-editor.org/rfc/rfc6605>.

[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>.

[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>.

[RFC7344] Kumari, W., Gudmundsson, O., and G. Barwood, "Automating DNSSEC Delegation Trust Maintenance", RFC 7344, DOI 10.17487/RFC7344, September 2014, <https://www.rfc-editor.org/rfc/rfc7344>.

[RFC7553] Faltstrom, P. and O. Kolkman, "The Uniform Resource Identifier (URI) DNS Resource Record", RFC 7553, DOI 10.17487/RFC7553, June 2015, <https://www.rfc-editor.org/rfc/rfc7553>.

[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>.

[RFC7766] Dickinson, J., Dickinson, S., Bellis, R., Mankin, A., and D. Wessels, "DNS Transport over TCP - Implementation Requirements", RFC 7766, DOI 10.17487/RFC7766, March 2016, <https://www.rfc-editor.org/rfc/rfc7766>.

[RFC7873] Eastlake 3rd, D. and M. Andrews, "Domain Name System (DNS) Cookies", RFC 7873, DOI 10.17487/RFC7873, May 2016, <https://www.rfc-editor.org/rfc/rfc7873>.

[RFC8078] Gudmundsson, O. and P. Wouters, "Managing DS Records from the Parent via CDS/CDNSKEY", RFC 8078, DOI 10.17487/RFC8078, March 2017, <https://www.rfc-editor.org/rfc/rfc8078>.

[RFC8080] Sury, O. and R. Edmonds, "Edwards-Curve Digital Security Algorithm (EdDSA) for DNSSEC", RFC 8080, DOI 10.17487/RFC8080, February 2017, <https://www.rfc-editor.org/rfc/rfc8080>.

[RFC8198] Fujiwara, K., Kato, A., and W. Kumari, "Aggressive Use of DNSSEC-Validated Cache", RFC 8198, DOI 10.17487/RFC8198, July 2017, <https://www.rfc-editor.org/rfc/rfc8198>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

[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>.

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <https://www.rfc-editor.org/rfc/rfc8555>.

[RFC8624] Wouters, P. and O. Sury, "Algorithm Implementation Requirements and Usage Guidance for DNSSEC", RFC 8624, DOI 10.17487/RFC8624, June 2019, <https://www.rfc-editor.org/rfc/rfc8624>.

[RFC8767] Lawrence, D., Kumari, W., and P. Sood, "Serving Stale Data to Improve DNS Resiliency", RFC 8767, DOI 10.17487/RFC8767, March 2020, <https://www.rfc-editor.org/rfc/rfc8767>.

[RFC8806] Kumari, W. and P. Hoffman, "Running a Root Server Local to a Resolver", RFC 8806, DOI 10.17487/RFC8806, June 2020, <https://www.rfc-editor.org/rfc/rfc8806>.

[RFC8976] Wessels, D., Barber, P., Weinberg, M., Kumari, W., and W. Hardaker, "Message Digest for DNS Zones", RFC 8976, DOI 10.17487/RFC8976, February 2021, <https://www.rfc-editor.org/rfc/rfc8976>.

[RFC9018] Sury, O., Toorop, W., Eastlake 3rd, D., and M. Andrews, "Interoperable Domain Name System (DNS) Server Cookies", RFC 9018, DOI 10.17487/RFC9018, April 2021, <https://www.rfc-editor.org/rfc/rfc9018>.

[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>.

[RFC9462] Pauly, T., Kinnear, E., Wood, C. A., McManus, P., and T. Jensen, "Discovery of Designated Resolvers", RFC 9462, DOI 10.17487/RFC9462, November 2023, <https://www.rfc-editor.org/rfc/rfc9462>.

[RFC9463] Boucadair, M., Ed., Reddy.K, T., Ed., Wing, D., Cook, N., and T. Jensen, "DHCP and Router Advertisement Options for the Discovery of Network-designated Resolvers (DNR)", RFC 9463, DOI 10.17487/RFC9463, November 2023, <https://www.rfc-editor.org/rfc/rfc9463>.

Acknowledgments

The authors thank Ross Gibson for his contributions, questions and comments.

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 Telecom
Email: roland.schott@telekom.de