Network Working Group
Internet Draft
Intended status: Informational
Expires: September 2, 2026

L. Dunbar
Futurewei
K. Majumdar
Oracle

March 2, 2026

Agent Attachment Protocol
draft-dunbar-agent-attachment-00

Abstract

This document describes the Agent Attachment Protocol (AAP), a network-centric mechanism that enables autonomous agents to securely attach to an overlay network infrastructure. The protocol defines procedures for agent attachment, identity validation, capability exchange, and secure session establishment between an agent and its local network edge node. Once attached, agents can communicate with remote agents through the overlay formed by interconnected edge nodes, without exposing the underlying network topology.

Status of this Memo

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on August 27, 2026 .

Copyright Notice

Table of Contents

1. Introduction.....3
2. Conventions used in this document.....4
3. Problem Statement.....5
4. Architecture Overview.....6
5. Agent Gateway Discovery and Attachment.....9
   5.1. Discovery Mechanisms.....9
      5.1.1. Anycast-Based Discovery.....9
      5.1.2. DHCP-Based Bootstrapping.....10
      5.1.3. DNS-Based Discovery via Directory Service.....10
   5.2. Offer and Selection.....10
   5.3. Handshake and Registration.....10
6. Messaging Forwarding and Routing.....11
   6.1. Transport Header.....11
   6.2. MFT Lookup and Forwarding.....11
7. Session Maintenance and Teardown.....12
   7.1. Stateful Attachment Model.....12
   7.2. Stateless (Soft-State) Attachment Model.....13
   7.3. Interoperability Considerations.....13
8. End-to-End Security.....13
9. Security Considerations.....14

10. Manageability Considerations.....14
11. IANA Considerations.....14
12. References.....14
12.1. Normative References.....14
12.2. Informative References.....14
13. Acknowledgments.....14
Appendix A:.....15

1. Introduction

The Agent Attachment Protocol (AAP) defines a network centric mechanism that separates agent application logic from network attachment and forwarding functions. In this architecture, agents are identified by stable, location independent identifiers, while network edge nodes provide scalable, policy-controlled forwarding based on those identifiers. Agents attach to a local edge node, which serves as a domain boundary and provides access to an overlay network formed by interconnected edge nodes.

AAP is limited to the attachment and access plane between an agent and its local network edge node. The protocol specifies procedures for agent discovery, secure attachment, authentication, capability exchange, session establishment, keepalive, and teardown. Once attached, agents communicate with remote agents via their respective network edge nodes, which forward messages based on transport-layer identifiers without inspecting, interpreting, or modifying application-layer content.

The protocol is agnostic to the application layer messaging format (e.g., A2A, MCP, or other frameworks) and does not mandate specific end to end security mechanisms between agents. Agent Gateways operate solely on AAP transport headers and forwarding state, abstracting underlying network topology while enabling scalable identity-based forwarding.

AAP supports operational requirements common to distributed agent systems, including mobility, multi-homing, and capability-based interaction. The protocol allows multiple bootstrap mechanisms for Agent to network edge discovery, including anycast based discovery, DHCP based provisioning,

and DNS based resolution. These discovery methods serve only as initial attachment mechanisms and do not alter the core attachment, authentication, and forwarding semantics defined in this document.

AAP does not define inter-node overlay control-plane procedures or application-layer messaging semantics. Its scope is strictly limited to secure network attachment and transport enablement for agent communications in multi-domain environments.

2. Conventions used in this document

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 BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The following acronyms and terms are used in this document:

Agent: An autonomous software entity identified by an Agent Identifier.

Agent Gateway (GW): A network edge node that functions as messaging routers within the agentic overlay network that manages the Message Forwarding Table (MFT) and handles bit-level transport for agents.

Directory Service: A capability registry that maps agent capabilities to the Agent ID.

Message Forwarding Table (MFT): A forwarding table mapping Agent ID prefixes to local interfaces or next-hop Gateways.

3. Problem Statement

The rapid emergence of AI-driven services and autonomous software agents has created increasing demand for agent-to-agent communication across administrative domains, cloud providers, and heterogeneous network environments. These agents may reside in enterprise networks, edge locations, or public cloud infrastructures, and often require secure, policy-controlled communication with remote agents outside their local domain.

Existing approaches typically rely on direct IP connectivity, application-layer messaging systems, or ad hoc overlay mechanisms. When agents span multiple domains, these approaches introduce several operational and architectural challenges. Direct connectivity models may expose internal endpoint addressing or topology information that operators prefer to abstract. There is no standardized mechanism to enforce forwarding or transport policies consistently across administrative boundaries. As the number of agents grows, maintaining direct connectivity models becomes operationally complex and difficult to scale. Additionally, current systems lack a common framework for secure agent attachment, identity validation, and gateway-mediated encrypted forwarding.

This document considers an architecture in which Agent Gateways (Agent GWs) serve as domain boundary nodes. Agents attach to a local Agent GW, and Agent GWs establish secure connectivity with other Agent GWs. Collectively, the interconnected Agent GWs form an overlay network that abstracts the underlying IP infrastructure, similar in concept to an SD-WAN overlay. In this model, end-to-end agent messages remain encrypted, while Agent GWs forward messages based on transport headers and policy controls without requiring visibility into application-layer semantics.

While emerging industry efforts (e.g., agent communication frameworks proposed by major cloud and networking vendors) demonstrate growing interest in agent-to-agent communication, many focus primarily on discovery, directory inquiry, or application-layer semantics. The problem addressed in this document is different: it focuses on defining a transport substrate between Agent Gateways. The goal is to provide a lightweight, secure, and policy-aware overlay messaging framework formed by Agent GWs that enables scalable

forwarding of encrypted agent-to-agent messages across domains.

The objective of this work is therefore to define an interoperable overlay transport protocol between Agent Gateways that supports secure agent attachment, controlled message forwarding, and domain-level policy enforcement, without mandating application-layer behavior or directory services.

4. Architecture Overview

Agent Gateway is a network edge node that functions as messaging routers within the agentic overlay network. They forward traffic based solely on information contained in the cleartext A2G transport header and local forwarding policy. Agent Gateways do not interpret, inspect, or modify application-layer payloads, and are not dependent on the semantics of the Agent-to-Agent (A2A) or MCP messaging carried within the transport.

The Agent-to-Gateway (A2G) protocol establishes a logical messaging overlay that decouples an agent’s identity and advertised capabilities from its physical network location. In this model, Agents are identified by stable AgentIDs, independent of IP addressing. In this model and the Agent Gateway (GW) acts as a specialized Messaging Router or Service Forwarder that anchors the agent’s current network attachment and provides scalable, identity-based forwarding across the overlay.

The architecture separates capability discovery from message delivery. A Directory Service provides "yellow-pages" functionality by mapping agent capabilities to stable Agent identifiers (AgentIDs). However, an agent’s network attachment may change over time due to mobility, multi-homing, policy decisions, or operational conditions, and therefore cannot be reliably inferred from capability information alone.

To address these dynamics, the A2G protocol defines a local discovery and handshake process in which an agent dynamically discovers an appropriate Agent Gateway, performs mutual authentication, and establishes a forwarding context. This attachment state is maintained at the Agent Gateway and reflected in its Message Forwarding Table (MFT), enabling

efficient routing based on AgentID while keeping discovery, identity, and forwarding responsibilities cleanly separated.

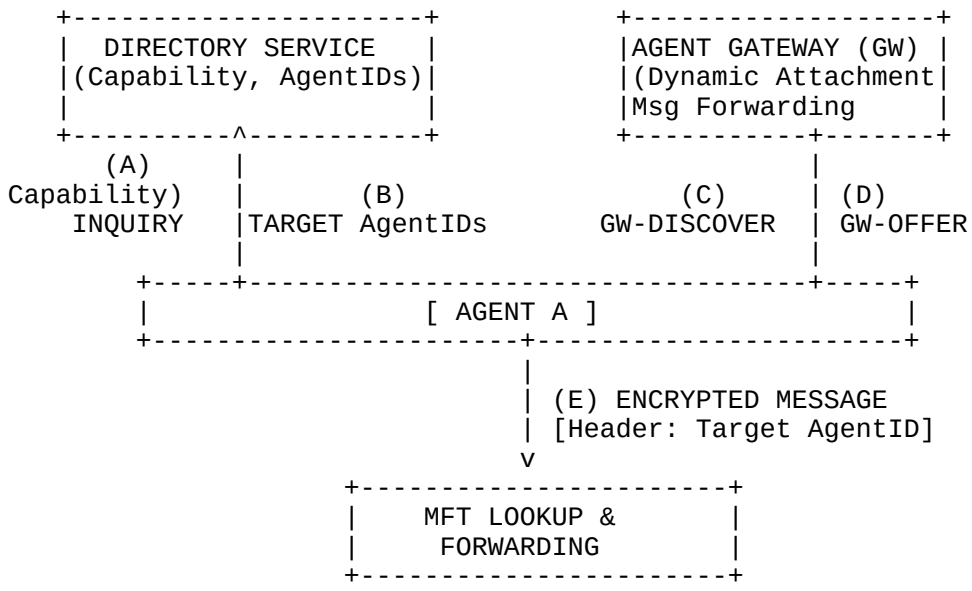

Figure 1: Key Components of Agent to GW Transport

Legend:

(A) Agent queries "Who has Skill X?"

(B) Directory returns "Agent B (AgentID-456)"

(C) Agent initiates an Agent Gateway discovery process to locate an appropriate local Agent Gateway that serves as its ingress point to the agentic overlay network.

(D) An Agent Gateway responds with a GW-OFFER, providing connection parameters and supported features. The Agent selects a Gateway and completes authentication and registration.

(E) Once attached, the Agent sends end-to-end messages containing a cleartext transport header with the Target AgentID and an application-layer payload. In most practical deployments, the A2A or MCP messaging carried within the payload is expected to be encrypted; however, this is independent of A2G-MTP. The Agent Gateway performs a lookup in its Message Forwarding Table (MFT) and

forwards the payload, without inspecting or interpreting its contents, toward the Gateway currently serving the target agent.

The following steps illustrate the high-level operation of the Agent-to-Agent Gateway transport:

- Capability Discovery:
  Agent performs an out-of-band inquiry to the Directory Service ((e.g., "find an agent capable of JSON-to-XML translation")). The Directory Service returns one or more Target Agent IDs that satisfy the request. At this stage, the Agent identifies which agent(s) it intends to communicate with, but does not yet have information about where those agents are currently attached in the network. The mechanism used for this capability inquiry is outside the scope of this document.

- Gateway Discovery:
  After identifying the target Agent ID(s), the Agent initiates an Agent Gateway discovery process by sending a GW-DISCOVER message (e.g., via anycast or other supported bootstrap mechanisms). One or more reachable Agent Gateways respond with GW-OFFER messages containing the Gateway identifier and connection parameters (such as a gRPC endpoint and supported features)

- Handshake and Registration:
  The Agent selects an Agent Gateway and performs a mutual authentication handshake, providing its Agent ID and cryptographic proof of identity. Upon successful validation, the Agent Gateway establishes an attachment context and registers the Agent in its local Message Forwarding Table (MFT).

- Message Forwarding and Table Management:
  Once attached, the Agent initiates messaging sessions by sending end-to-end payloads. Each message includes a cleartext A2G transport header that carries the Source Agent ID and Target Agent ID.

  Upon receiving a message, the Agent Gateway does not inspect the payload. Instead, it performs a lookup on the Target Agent ID in its MFT. If a matching entry exists, the Gateway forwards the message to the

appropriate local interface or next-hop Agent Gateway. If no entry is found, the Gateway may invoke a route-discovery procedure (outside the scope of this document) to locate the Gateway currently serving the target agent.

The MFT is updated dynamically as agents attach, detach, or move, reflecting the evolving attachment state of the agentic overlay network.

5. Agent Gateway Discovery and Attachment

In a dynamic environment, an Agent cannot assume a static configuration for its messaging ingress. An Agent's network attachment may change over time due to mobility, multi-homing, policy decisions, or deployment constraints. Therefore, A2G-MTP defines a Gateway discovery and attachment procedure that allows an Agent to identify an appropriate Agent Gateway and establish a forwarding context.

This document supports multiple mechanisms for discovering an Agent Gateway. These mechanisms are treated as bootstrap options and do not alter the attachment, authentication, or forwarding semantics defined in subsequent sections.

5.1. Discovery Mechanisms

An Agent MAY use one or more of the following mechanisms to discover a candidate Agent Gateway:

5.1.1. Anycast-Based Discovery

An Agent MAY discover a local ingress Gateway by sending a GW-DISCOVER message to a well-known routable anycast address. The discovery message includes the Agent's ID and a nonce.

Any Agent Gateway reachable via the anycast address MAY respond with a GW-OFFER message. Anycast-based discovery enables topology-aware selection of a nearby Gateway without requiring pre-configuration.

5.1.2. DHCP-Based Bootstrapping

In managed access networks, the Agent Gateway MAY be co-located with, or designated by, the IP gateway for the Agent's access network. In such environments, DHCP MAY be used to provide the Agent with unicast bootstrap information for an Agent Gateway, such as an IP address, FQDN, or service endpoint.

DHCP-based information MUST be treated as advisory. The Agent MUST still perform the authentication and registration procedures defined in Section 4.3 before establishing an attachment.

5.1.3. DNS-Based Discovery via Directory Service

The Directory Service MAY return a stable Agent Gateway service identifier (e.g., URL or FQDN) to the Agent as part of the capability discovery process. The Agent MAY use DNS resolution (including anycast or load-balanced DNS) to obtain a reachable Agent Gateway unicast address.

DNS-based discovery provides scalable, multi-hop reachability but does not replace the explicit attachment and authentication procedures defined by A2G-MTP.

5.2. Offer and Selection

For discovery mechanisms that involve Gateway responses (e.g., anycast-based discovery), an Agent Gateway responds with a GW-OFFER message containing:

- Gateway identifier
- Transport endpoint (e.g., gRPC URI)
- Supported transport features

The Agent selects one Gateway based on local policy and proceeds with the attachment procedure.

5.3. Handshake and Registration

The Agent initiates a mutual authentication handshake with the selected Agent Gateway, providing its Agent ID and cryptographic proof of identity. Upon successful validation, the Agent Gateway establishes an attachment context and assigns a Session Context, analogous to a DHCP lease.

The Agent Gateway then registers the Agent in its local Message Forwarding Table (MFT), enabling message forwarding for the duration of the attachment.

Regardless of the discovery mechanism used, the handshake, authentication, registration, and forwarding behavior defined by A2G-MTP remain identical.

6. Messaging Forwarding and Routing

The Agent GW serves as the switching and forwarding engine for the agentic overlay network. Each Agent Gateway maintains a MFT that maps Agent IDs (or Agent ID prefixes) to local attachment or next-hop Agent Gateways. The MFT reflects the dynamic attachment state of agents and is updated as agents attach, detach, or move.

6.1. Transport Header

Each message sent by an Agent carries a cleartext A2G transport header that enables forwarding without exposing application semantics. The transport header includes the following fields:

- Target Agent ID (or prefix)
- Message Identifier
- TTL
- End-to-End Security Context Identifier

In most deployments, application data and higher-layer semantics are encrypted end-to-end between agents and are opaque to Agent Gateways.

6.2. MFT Lookup and Forwarding

Ingress Processing: When an Agent sends a message, it includes a clear-text A2G Header containing the Target Agent ID, which was obtained earlier from the capability discovery. The encrypted payload is encapsulated without modification by the Agent.

Forwarding Logic: Upon receiving a message, the Agent Gateway performs an exact-match (or prefix-match, if supported) lookup on the Target Agent ID in its MFT.

- If the target is attached to the same Gateway, the Gateway forwards the encrypted payload to the corresponding local attachment.
- If the target is attached to a remote Gateway, the Gateway forwards the encrypted payload to the next-hop Agent GW.

If no matching MFT entry exists, the Gateway MAY invoke a route-discovery or resolution procedure to locate the Gateway currently serving the target Agent; such procedures are outside the scope of this document.

Payload Confidentiality: Agent GW does not inspect, decrypt, or interpret the message body. The GW only forwards messages solely based on the information contained in the cleartext transport header.

7. Session Maintenance and Teardown

The relationship between the Agent and the GW may be stateful or stateless (soft state), depending on deployment requirements and implementation choices. In both cases, the attachment is ephemeral and reflects the Agent’s current network presence.

7.1. Stateful Attachment Model

In a stateful attachment model, the Agent Gateway maintains explicit session state for each attached Agent.

Keep-Alives: An attached Agent periodically sends "Heartbeat" messages to the GW to indicate continued presence. If heartbeats are not received within a configured interval, the GW marks the corresponding MFT entry as Stale and eventually remove it.

Explicit Release: When an Agent finishes its task or intends to detach, it sends a Session-End signal to the Gateway. Upon receipt, the Gateway removes the Agent’s entry from its MFT and MAY notify Gateways to tear down the corresponding forwarding state.

7.2. Stateless (Soft-State) Attachment Model

In a soft-state attachment model, the Gateway does not rely on explicit session teardown and instead treats attachment state as time-bounded.

- MFT entries are created with a lifetime and expire automatically unless refreshed.
- Periodic data traffic or lightweight refresh messages MAY implicitly renew the attachment.
- Absence of traffic or refreshes results in automatic removal of stale MFT entries without requiring explicit Session-End signaling.

The soft-state model reduces signaling overhead and is well suited for large-scale or highly dynamic environments where explicit session management may be impractical.

7.3. Interoperability Considerations

Regardless of whether a stateful or soft-state model is used, the forwarding behavior defined in Section 5 remains unchanged. Gateways forward messages based on current MFT entries and do not depend on application-layer semantics.

Implementations MAY support one or both attachment models, provided that stale attachment state is removed in a timely manner to prevent mis-delivery.

8. End-to-End Security

A2G-MTP is agnostic to the application-layer messaging protocol used between Agents (e.g., A2A, MCP) and does not mandate whether Agent-to-Agent communications are encrypted. In most practical deployments, such communications are expected to be protected using end-to-end encryption.

When end-to-end security is employed, the selected mechanism is responsible for providing confidentiality, integrity protection, replay protection, and sender authentication between communicating Agents.

Agent Gateways operate solely on the cleartext A2G transport header for forwarding decisions. Agent Gateways MUST NOT

decrypt, inspect, or modify application layer payloads and are not participants in application-layer security contexts.

The specific end-to-end security protocol used between Agents is outside the scope of this document. Any selected mechanism SHOULD provide mutual authentication and cryptographic binding between the Agent identity and the protected message content.

9. Security Considerations

TBD.

10. Manageability Considerations

TBD

11. IANA Considerations

TBD

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

12.2. Informative References

.

13. Acknowledgments

Acknowledgements to xxx for their extensive review and suggestions.

This document was prepared using 2-Word-v2.0.template.dot.

Appendix A:

Authors’ Addresses

Linda Dunbar
Futurewei
Email: ldunbar@futurewei.com

Kausik Majumdar
Oracle
Email: kausik.majumdar@oracle.com

Contributors’ Addresses