Network Working Group
Internet Draft
Intended status: Informational
Expires: November 1, 2026

L. Dunbar
Futurewei
K. Majumdar
Oracle

May 1, 2026

Agent Attachment Protocol
draft-dunbar-agent-attachment-01

Abstract

This document describes the Agent Attachment Protocol (AAP) that enables AI agents to establish attachment to an edge node and derive communication and attachment-related properties. These properties include endpoint identifiers, supported communication mechanisms, and attachment context information that can be exposed to discovery systems. AAP focuses on how agents obtain and maintain attachment state and how attachment-derived properties can be represented in a consistent and interoperable manner. These properties can be used by forwarding functions at edge nodes and by routing or control-plane mechanisms to support communication between agents.

Status of this Memo

This Internet-Draft will expire on November 1, 2026 .

Copyright Notice

Table of Contents

1. Introduction

The Agent Attachment Protocol (AAP) defines a mechanism for agents to establish attachment to an edge node and to derive communication and attachment-related properties associated with that attachment.

Some AI agents, such as those managed by OpenClaw, operate in cloud environments, while others run on devices connected through private networks, such as smart-factory control agents and AI agents hosted on user equipment (UE) in 5G/6G systems.

Those device-hosted agents are typically associated with physical devices or edge infrastructure, such as industrial equipment, vehicles, sensors, or user equipment (UE), and operate within a controlled administrative domain where:

- connectivity is restricted and not globally reachable,

- o naming and addressing may be private or domain-specific, and
- o security and policy enforcement depend on local network context.

In these settings, an agent is identified by a stable, location-independent identifier (e.g., AgentID), but its ability to communicate depends on its current network attachment. An agent may attach to different access routers, gateways, or edge nodes over time due to mobility, scaling, failover, or operational policies.

As a result, static endpoint information or identifiers alone are insufficient to support efficient and policy-compliant communication. Instead, it is necessary to understand where and how an agent is currently attached, including its attachment point, access domain, and associated communication properties.

AAP addresses this gap by defining how agents:

- o attach to a local network edge node,
- o establish an attachment context, and
- o derive communication properties associated with that attachment.

These attachment derived properties may include endpoint identifiers, supported protocols, attachment point identifiers, and domain information. Such properties can be made available to other systems, such as discovery services, forwarding functions, or control-plane mechanisms, to support efficient, secure, and policy-aware communication.

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:

AI Agent: A software entity that uses artificial intelligence techniques to interpret context, make decisions, and take actions toward achieving one or more goals, possibly by interacting with users and by invoking other agents, tools, or services, rather than only executing fixed pre-programmed functions.

Attachment Context: The set of properties describing an agent's current network attachment (e.g., attachment point, domain, protocols).

Agent Gateway (GW): An edge node that serves as an attachment anchor for an Agent within a provider-managed or private network.

Attachment State: The maintained representation of an agent's attachment context at an Agent Gateway.

Directory Service: A service or registry that enables discovery of agents based on their capabilities and returns information associated with those agents, such as Agent Identifiers (AgentIDs), endpoints, or metadata. One example is a registry based on A2A Agent Cards, where an agent publishes a card describing its capabilities, skills, and endpoint.

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

3. Problem Statement

Some AI agents operate on devices or edge infrastructure connected through private networks, such as smart-factory control agents and AI agents hosted on user equipment (UE) in 5G/6G systems. In these environments, communication depends not only on an agent's identity, but also on its current network attachment.

These environments typically have limited reachability, private or domain-specific naming and addressing, and security and policy requirements tied to local network context. As a result, an agent identifier alone (e.g., AgentID) is often insufficient to determine how the agent can be reached or how it should interact with other entities. Effective communication depends on attachment-related context, such as the access domain, attachment point, and associated communication properties.

An agent’s attachment may change over time due to mobility, scaling, failover, or operational policies. Existing discovery systems, such as services based on A2A Agent Cards, can identify entities and return basic information, such as identifiers and capability descriptions, but they typically lack a standardized way to represent and expose attachment-related properties that reflect where and how an agent is currently attached within a given administrative domain.

Without a consistent mechanism for deriving and representing such properties, discovery information may be incomplete or outdated, communication may be suboptimal or fail because of topology or policy constraints, and control-plane mechanisms lack a common basis for exchanging attachment-related information across network elements.

This document addresses the need for a standardized mechanism by which agents can derive and expose attachment-related communication properties that reflect their current network attachment within a controlled administrative domain. These properties can then be made available to other systems, such as discovery services or control-plane mechanisms, to support efficient, policy-compliant, and secure communication.

4. Use Cases

This section provides illustrative examples of agents that are attached to edge nodes within provider-managed environments, highlighting how they differ from conventional application deployments and why attachment-related properties are required.

4.1. Smart Factory Control Agent

In a smart factory environment, industrial robots and control systems operate within a standalone private network managed by the factory operator.

An AI agent associated with a worker’s AR (Augmented Reality) glasses may assist with inspection, maintenance, or repair tasks by invoking multiple tools or services available within the factory network. For example, the agent may:

- retrieve telemetry from nearby machines or sensors,
- invoke a diagnostic tool hosted on a local edge server,
- query an inventory or parts database,
- obtain repair procedures from an enterprise application, and
- request real-time visual or language assistance from a local inference service.

To complete such tasks, the agent may need to discover and interact with services that are available only within the factory domain and reachable through the worker’s current point of attachment. The agent’s communication and behavior may therefore depend on the specific gateway, edge node, or factory segment to which the AR glasses are connected.

Unlike traditional industrial applications that rely on fixed workflows or centralized control systems, such an agent can dynamically select and invoke different tools or services based on the worker’s task, location, and surrounding conditions. As the worker moves across the factory floor or connects through a different access point or edge node, the agent’s reachable services and communication properties may also change.

4.2. Private 5G/6G Service Agent

In a private 5G/6G deployment, an enterprise or operator may provide a controlled network domain to support localized and latency-sensitive services.

An AI agent associated with user equipment (UE), such as a field technician tablet, autonomous vehicle terminal, or industrial handheld device, may need to accomplish a complex

task by invoking multiple remote tools or services hosted within the private network. For example, the agent may:

- o retrieve equipment status from a monitoring service,
- o invoke a diagnostic tool hosted on an edge MCP server,
- o access a local policy or authorization service,
- o query a nearby inference service for real-time analysis, and
- o obtain repair instructions or workflow data from an enterprise application server.

To complete such tasks efficiently, the agent may need to discover and interact with different services available at the current site or within the serving 5G/6G domain. Which services are reachable, and how they should be accessed, can depend on the UE’s current point of attachment, serving edge node, network slice, or operator policy.

Unlike typical Internet-based applications that rely on globally reachable endpoints, such agents operate within a provider-managed domain where communication depends on the current network attachment and locally available services. As the UE moves, changes serving edge nodes, or is reassigned to a different slice or policy domain, the agent’s reachable resources and communication properties may also change.

4.3. Industrial IoT Monitoring Agent

In industrial IoT deployments, sensors and devices are connected through private access networks and edge gateways.

An AI agent associated with a device or gateway may monitor local conditions and respond to operational events by invoking multiple tools or services within the local domain. For example, the agent may:

- o collect telemetry from nearby sensors and controllers,
- o invoke an anomaly-detection or correlation service hosted on an edge server,
- o query a local policy or safety system to determine permitted actions, and
- o trigger alerts, workflow updates, or local control responses.

Unlike traditional monitoring systems that periodically export data to centralized cloud platforms for offline analysis, such agents perform local analysis and decision-making within the private network. Their ability to reach the appropriate tools or services may depend on the device's current attachment point, serving gateway, or local policy domain.

4.4. Key Characteristics

The above examples share the following properties:

- Agents may need to query a Directory Service to discover other agents, tools, or services needed to complete a task.
- Agents may invoke remote services, such as tools hosted on MCP servers or other service endpoints within the local or remote domain.
- Agents may invoke or coordinate with other agents to accomplish multi-step tasks.
- Agents are attached to edge nodes, gateways, or physical devices.
- They operate within controlled administrative domains rather than solely over the public Internet.
- Their behavior depends on local context, including device state, locally available services, policy constraints, and current network attachment.
- Their attachment may change over time, requiring dynamic updates to communication properties and reachable resources.

In such environments, an agent's identifier alone is insufficient to determine how it can be reached or what resources it can access. The agent's current attachment context becomes an essential part of its operational state.

5. Architecture Overview

The Agent Attachment Protocol (AAP) defines the procedures by which an agent establishes attachment to an edge node, referred to as an Agent Gateway.

An Agent Gateway is an edge node that serves as the attachment anchor for agents. It provides functions for agent authentication, registration, and maintenance of attachment state. Gateway anchors the agent’s current network presence and maintains an association between the Agent Identifier (AgentID) and its point of attachment.

In this architecture, agents are identified by stable, location-independent AgentIDs, decoupled from underlying IP addressing. Agent Gateway establishes and maintains the attachment context for each agent, which includes information such as the agent’s current ingress point, associated domain or policy context, and supported communication mechanisms.

The primary function of AAP is to create and maintain this attachment context. The protocol defines how an agent:

- identifies and connects to a candidate Agent Gateway,
- performs mutual authentication,
- registers its AgentID, and
- establishes and maintains its attachment state.

As part of this process, the Agent Gateway may derive a set of attachment-related communication properties associated with the agent’s current attachment. These properties can include endpoint identifiers, supported protocols, attachment point identifiers, and domain information.

The communication properties derived from attachment may be exposed to external systems, such as discovery services, or used by other mechanisms (e.g., routing or control-plane protocols). However, the definition of those systems and their use of the information is outside the scope of this document.

While an Agent Gateway may participate in forwarding traffic based on local policy or identifiers, forwarding behavior and overlay control-plane mechanisms are not specified by AAP; examples of how attachment state may be used by such mechanisms are described in Section 7.

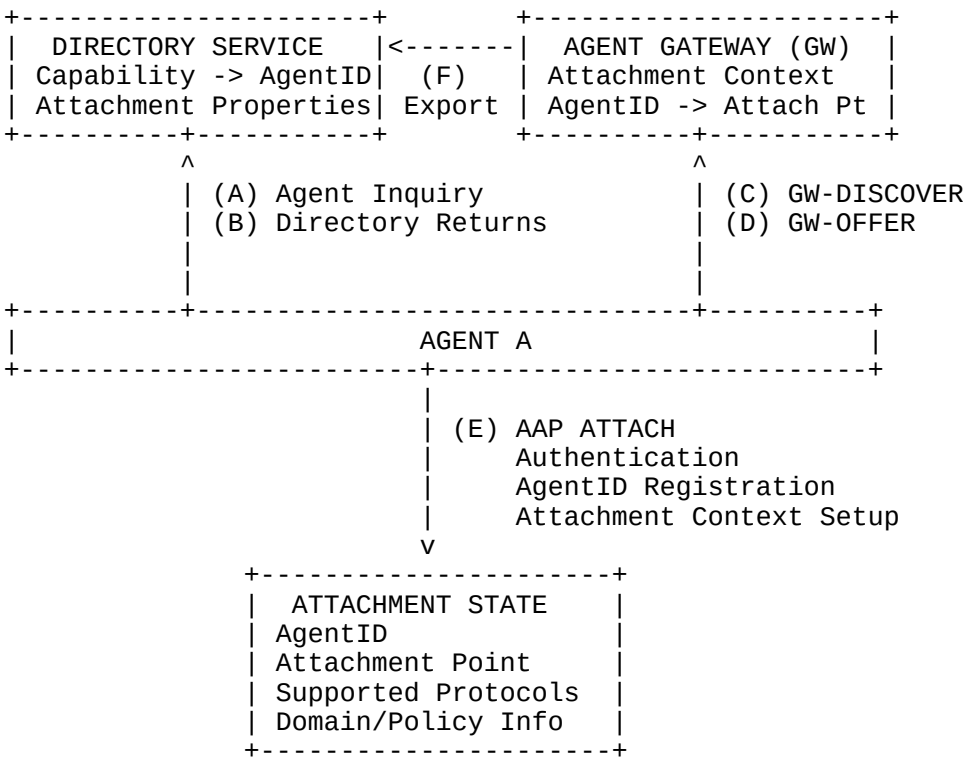

Figure 1: Key Components of Agent Attachment

Legend:

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

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

(C) GW-DISCOVER: The Agent initiates discovery of a candidate Agent Gateway using locally configured or bootstrap mechanisms (e.g., anycast, DNS, or pre-configured address). This step is part of attachment bootstrapping and is within the scope of AAP.

(D) GW-OFFER: An Agent Gateway responds with information needed for attachment, such as its identifier, reachable endpoint, and supported attachment parameters. The Agent selects a Gateway and proceeds with attachment.

(E) AAP ATTACH: The Agent establishes attachment to the selected Agent Gateway by performing mutual authentication,

registering its AgentID, and creating an attachment context. The Gateway records the association between the AgentID and the current attachment point, and derives attachment-related communication properties (e.g., endpoint, supported protocols, domain/policy context).

(F) Attachment Property Exposure: The Agent Gateway, or another authorized attachment authority, may expose attachment-derived properties to a Directory Service or similar system. These properties enable discovery systems to return information about how and where an entity is currently attached. The mechanisms for publishing, distributing, or querying this information are outside the scope of AAP.

The following steps illustrate the high level operation of agent attachment to an Agent Gateway:

- Capability Discovery: An Agent may perform an out-of-band inquiry to a Directory Service, for example, to find an agent capable of JSON-to-XML translation. The Directory Service may return one or more Target AgentIDs that satisfy the request. At this stage, the Agent has identified the target agent(s) but may not know where those agents are currently attached in the network. The capability discovery mechanism is outside the scope of this document.

- Gateway Discovery: The Agent discovers a candidate local Agent Gateway using a supported bootstrap mechanism, such as anycast, DNS-based resolution, DHCP-based provisioning, or pre-configuration. The Agent may send a GW-DISCOVER message, and one or more reachable Agent Gateways may respond with GW-OFFER messages containing the Gateway identifier, attachment endpoint, supported protocols, and other attachment parameters.

- Handshake and Registration: The Agent selects an Agent Gateway and performs a mutual authentication handshake. During this process, the Agent provides its AgentID and cryptographic proof of identity. Upon successful validation, the Agent Gateway establishes an attachment context and records the association between the AgentID and the current attachment point.

- Attachment Property Derivation: After successful attachment, the Agent Gateway derives and maintains attachment related communication properties for the Agent. These may include endpoint identifiers, supported protocols, attachment point identifiers, domain information, and policy context. These properties reflect the Agent's current network attachment and may change as the Agent moves, detaches, re-attaches, or changes Gateway.

- Exposure to External Systems: The attachment-derived properties may be made available to external systems, such as Directory Services or future control-plane mechanisms. This enables discovery systems to return information about how and where an Agent is currently attached. The mechanisms for publishing, querying, distributing, or using these properties are outside the scope of this document.

6. Agent Gateway Bootstrap and Attachment

In a dynamic environment, an Agent cannot assume a fixed attachment point. An Agent's network attachment may change over time due to mobility, multi-homing, scaling, policy decisions, or deployment constraints. Therefore, the Agent Attachment Protocol (AAP) defines bootstrap mechanisms that allow an Agent to identify a candidate Agent Gateway and establish an attachment context.

The mechanisms described in this section are limited to identifying a candidate Agent Gateway for attachment and establishing the attachment state. They are not part of entity discovery and do not define how agents discover other agents.

AAP supports multiple mechanisms for locating a candidate Agent Gateway. These mechanisms are treated as bootstrap options and do not alter the attachment, authentication, or registration procedures defined in this document.

6.1. Bootstrap Mechanisms

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

6.1.1. Anycast-Based Discovery

An Agent MAY identify a nearby Agent Gateway by sending a GW-DISCOVER message to a well-known anycast address. The message MAY include the AgentID and a nonce.

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

6.1.2. 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 5.3 before establishing an attachment.

6.1.3. DNS-Based Bootstrap

A system external to AAP (e.g., a directory or configuration service) MAY provide a stable Agent Gateway identifier, such as a URL or FQDN. The Agent MAY use DNS resolution to obtain a reachable address for the Gateway.

DNS-based bootstrap provides scalable reachability but does not replace the explicit attachment and authentication procedures defined by AAP..

6.2. Gateway Offer and Selection

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

- Gateway identifier
- Transport endpoint (e.g., URI or IP address)
- Supported attachment protocols and parameters

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

6.3. Handshake and Registration

The Agent initiates a mutual authentication handshake with the selected Agent Gateway, providing its AgentID and cryptographic proof of identity.

Upon successful validation, the Agent Gateway establishes an attachment context and records the association between the AgentID and the current attachment point. This attachment context reflects the Agent’s current network presence and may include information such as attachment point identifier, supported protocols, and domain or policy context.

The Agent Gateway maintains this attachment state for the duration of the attachment.

6.4. Attachment Property Derivation

Following successful attachment, the Agent Gateway derives and maintains a set of attachment-related communication properties associated with the Agent’s current attachment.

These properties MAY include:

- Endpoint identifiers
- Supported communication protocols
- Attachment point identifiers
- Domain or administrative context
- Policy-related attributes

These properties reflect the Agent’s current network attachment and may change as the Agent moves, detaches, or re-attaches.

6.5. Exposure of Attachment Properties

The attachment-related properties derived by the Agent Gateway MAY be made available to external systems, such as directory services or other control-plane mechanisms.

This enables discovery systems to include information about how and where an Agent is currently attached. The mechanisms for publishing, distributing, querying, or consuming these properties are outside the scope of AAP.

7. Use of Attachment State (Informative)

This section describes how attachment state established by AAP may be used by other mechanisms. The behavior described here is illustrative only. AAP does not define forwarding protocols, routing mechanisms, or transport architectures, but provides the attachment-derived information required to enable them.

7.1. Attachment State Representation

An Agent Gateway maintains attachment state for each attached Agent. This state reflects the current binding between an AgentID and its attachment context.

The attachment state MAY include:

- AgentID
- Attachment point identifier (e.g., local interface or ingress context)
- Reachable endpoint identifier (e.g., URI, IP address, or service endpoint)
- Supported communication protocols
- Domain or administrative context
- Policy-related attributes

This information is derived during the attachment process and updated as the Agent attaches, detaches, or changes its point of attachment.

7.2. Use of Attachment State by External Systems

Attachment state may be consumed by systems external to AAP, such as:

- discovery or directory services,
- routing or control-plane mechanisms,
- policy enforcement systems.

These systems may use attachment related properties to determine how to reach an Agent, exchange reachability information, or optimize communication paths.

For example, a control-plane mechanism may use attachment point identifiers and domain information to select a preferred path to an Agent when multiple attachment options exist.

AAP does not define how such systems operate or how the information is exchanged between them.

7.3. Forwarding Using Attachment State (Example)

In some deployments, attachment state may be used by edge nodes to support message forwarding between Agents.

An Agent Gateway may act as a forwarding element within a domain or overlay system. Each Gateway MAY maintain a data structure (e.g., a Message Forwarding Table, MFT) that maps AgentIDs (or AgentID prefixes) to local attachment points or next-hop Agent Gateways.

The MFT is derived from attachment state and updated dynamically as agents attach, detach, or change their point of attachment.

When a message is sent toward a target AgentID:

- If the target Agent is attached locally, the message may be delivered to the corresponding attachment point.

- If the target Agent is attached to a remote Gateway, the message may be forwarded to a next-hop Gateway based on locally available state or control-plane information.

If no matching entry exists, the Gateway MAY invoke a resolution or discovery procedure to determine the current attachment point of the target Agent. Such procedures are outside the scope of this document.

7.4. Transport Header (Example)

AAP facilitates forwarding by enabling the use of forwarding-relevant identifiers derived from attachment state.

In an example deployment, messages MAY include a cleartext transport header carrying information such as:

- Target AgentID (or prefix)
- Message Identifier
- Time-to-Live (TTL)
- End-to-End Security Context Identifier

This header enables forwarding decisions based on AgentID while keeping application-layer semantics opaque to intermediate nodes.

7.5. Role in Routing and Control Plane

Attachment-derived properties defined by AAP may also be used by routing or control-plane mechanisms.

For example:

- control-plane systems may exchange AgentID-to-attachment mappings across domains,
- routing mechanisms may use attachment point identifiers and domain information to determine optimal paths, and
- policy systems may use attachment context to enforce domain-specific constraints.

AAP does not define these routing or control-plane mechanisms, but provides the necessary attachment information to enable them.

7.6. Payload Confidentiality

Agent Gateways do not inspect, decrypt, or interpret application-layer payloads. Forwarding decisions are made solely based on transport header information and attachment-derived state.

End-to-end confidentiality and integrity are preserved between communicating agents.

8. Attachment State Maintenance and Teardown

The relationship between an Agent and an Agent Gateway is represented by attachment state, which reflects the Agent’s current network presence. This attachment state is dynamic and may change over time as the Agent attaches, detaches, or changes its point of attachment.

Attachment state MAY be maintained using either a stateful model or a soft-state (stateless) model, depending on deployment requirements and implementation choices.

8.1. Stateful Attachment Model

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

Keep-Alives:

An attached Agent periodically sends keep-alive or heartbeat messages to the Agent Gateway to indicate continued presence. If these messages are not received within a configured interval, the Agent Gateway MAY mark the corresponding attachment state as stale and eventually remove it.

Explicit Release:

When an Agent intends to detach, it MAY send a termination signal (e.g., Session-End) to the Agent Gateway. Upon receipt, the Agent Gateway removes the attachment state associated with the AgentID.

This model provides explicit control over attachment lifecycle and enables precise tracking of active attachments.

8.2. Stateless (Soft-State) Attachment Model

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

- Attachment state entries are created with a lifetime and expire automatically unless refreshed.
- Periodic refresh messages or ongoing communication MAY implicitly renew the attachment state.
- Absence of refresh activity results in automatic removal of stale attachment state.

This model reduces signaling overhead and is suitable for large-scale or highly dynamic environments.

8.3. Interoperability Considerations

Regardless of whether a stateful or soft-state model is used, the semantics of attachment state remain consistent. Implementations MUST ensure that stale or invalid attachment state is removed in a timely manner to avoid incorrect or outdated representation of an Agent’s network attachment.

Implementations MAY support one or both models, provided that the resulting attachment state accurately reflects the Agent’s current attachment context.

9. Security Considerations

The Agent Attachment Protocol (AAP) defines procedures for establishing attachment between an Agent and an Agent Gateway. Security in AAP focuses on protecting the integrity of the attachment process and the correctness of the resulting attachment state.

9.1. Authentication and Identity Binding

AAP MUST support mutual authentication between the Agent and the Agent Gateway during the attachment process.

- The Agent MUST provide cryptographic proof of its identity (e.g., possession of a private key associated with its AgentID).
- The Agent Gateway MUST authenticate the Agent before establishing attachment state.
- The Agent MUST validate the identity of the Agent Gateway to prevent attachment to unauthorized or malicious gateways.

The attachment context established by AAP SHOULD include a cryptographic binding between the AgentID and the authenticated session to prevent impersonation or replay.

9.2. Integrity of Attachment State

The Agent Gateway maintains attachment state that associates an AgentID with a current attachment point and related properties.

Implementations MUST ensure that:

- attachment state cannot be modified by unauthorized entities,
- stale or invalid attachment state is removed in a timely manner, and
- updates to attachment state are authenticated and integrity protected.

Incorrect or compromised attachment state may result in misrepresentation of an Agent’s reachability or communication properties.

9.3. Protection of Attachment Exchanges

Messages exchanged during bootstrap, authentication, and attachment establishment SHOULD be protected against eavesdropping, tampering, and replay.

This MAY be achieved using existing secure transport mechanisms (e.g., TLS, DTLS, QUIC, or equivalent). The selection of specific transport protocols is outside the scope of this document.

9.4. Exposure of Attachment Properties

Attachment-related communication properties derived by the Agent Gateway may be exposed to external systems, such as directory services.

Care MUST be taken to ensure that:

- only authorized systems can access attachment-related information,
- sensitive information (e.g., internal topology or policy context) is not exposed beyond intended scope, and
- integrity and authenticity of published properties are preserved.

Mechanisms for access control, distribution, and validation of such information are outside the scope of AAP.

9.5. Relationship to End-to-End Security

AAP does not define agent-to-agent communication or application-layer messaging. Therefore:

- AAP does not mandate end-to-end encryption between Agents.
- AAP does not participate in application-layer security contexts.

End-to-end security between Agents (e.g., confidentiality, integrity, and authentication of exchanged data) MUST be provided by mechanisms external to AAP.

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