

Agent-GW
Internet-Draft
Intended status: Standards Track
Expires: 2 September 2026

Xiaohui. Xie
Tsinghua University
Zian. Wang
Beijing University of Posts and Telecommunications
Tianshuo. Hu
Yong. Cui
Tsinghua University
1 March 2026

Agent Communication Gateway for Semantic Routing and Working Memory
draft-agent-gw-01

Abstract

This document presents an architectural framework for an Agent Communication Gateway (Agent-GW), designed to support large-scale, heterogeneous, and dynamic multi-agent collaboration across administrative and protocol boundaries.

As agents evolve from isolated entities to a collaborative digital workforce, the infrastructure must transition from rigid, endpoint-based connectivity to intent-based interaction. This draft proposes Agent-GW as an infrastructure hub that provides native primitives for Semantic Routing (dispatching tasks by intent and capability), Working Memory (shared structured context across multi-step workflows), automated protocol adaptation (normalizing heterogeneous interfaces into a unified agent-facing protocol), oracle-free agent evaluation, and collaborative inference acceleration via a Knowledge Delivery Network (KDN).

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 2 September 2026.

Copyright Notice

Copyright (c) 2026 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
2. Conventions used in this document	3
3. Terminology	3
4. Deployment Model and Trust Boundary	4
5. Network and Infrastructure Requirements	5
6. Architecture Overview	5
6.1. Internal/External Entity Relationship	6
6.2. Functional Planes	7
7. Protocol Model and Input/Output Examples	7
7.1. Example: A2A Request Normalization	7
7.2. Example: Southbound Output Mapping	8
8. Infrastructure Functions	8
8.1. Agent Identification and Capability Directory	8
8.2. Automated Protocol Adaptation and Interface Normalization (APA)	9
8.3. Infrastructure-Level Agent Evaluation and Compliance	9
8.4. Dynamic Orchestration and Semantic Routing	9
8.5. Evolutionary Knowledge Management	9
8.6. Collaborative Inference Acceleration (KDN)	10
9. Representative Deployment Scenarios	10
9.1. Scenario 1: Enterprise Copilot with Local Secure Routing and External Egress	10
9.2. Scenario 2: Industrial Planning with IoT (MQTT/REST) and Embodied Agent (ROS Bridge)	11
9.3. Scenario 3: Peer Agent-GW Synchronization and Inference Artifact Transfer	12
10. Security Considerations	12
11. IANA Considerations	13
12. Acknowledgements	13
13. References	13
13.1. Normative References	13
Authors' Addresses	13

1. Introduction

The rapid advancement of Large Language Models (LLMs) has catalyzed the emergence of an "Internet of Agents", where autonomous software entities and tool-like services interconnect to form collaborative workflows. Unlike traditional microservices, agents have varying degrees of autonomy, reasoning capabilities, and diverse interface standards. Early deployments were often siloed within proprietary frameworks, limiting cross-domain collaboration.

As these systems scale, the bottleneck shifts from basic connectivity to context management and efficient orchestration. Delivering the right context to the right agent at the right time, while controlling the cost of inference, becomes an infrastructure challenge. Existing gateways optimized for static endpoints and stateless forwarding lack semantic awareness to interpret intents or manage multi-step task lifecycles.

This document introduces the Agent Communication Gateway (Agent-GW), situated between agents and external tools or services. Agent-GW elevates the network from a passive transport layer to an active semantic intermediary by introducing two core primitives: Semantic Routing (intent/capability-based dispatch) and Working Memory (shared, incrementally updated context). It further defines protocol adaptation, evaluation, observability, and KDN-based inference acceleration.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

The following terms are defined in this draft:

Agent-GW (Agent Communication Gateway) An infrastructure component coordinating multi-agent communication, responsible for protocol adaptation, semantic routing, and context management.

Internal Semantic Domain (ISD) The internal network domain, typically a LAN or on-prem cluster, where agent-to-agent messages follow standardized or controlled protocols (e.g., A2A, MCP, or natural language over a controlled channel). The ISD is considered within a primary trust boundary.

External Heterogeneous Ecosystem (EHE) External networks and services outside the LAN trust boundary, often with diverse and unstructured protocols and varying security postures (e.g., public SaaS APIs, Internet services, third-party tools).

Semantic Routing Routing a request based on the semantic intent of a task and the capabilities/trust state of available agents, rather than static endpoint addresses.

Working Memory A structured, temporary storage mechanism that maintains context and state across a multi-step or multi-turn workflow. It can be session-scoped and policy-controlled.

KDN (Knowledge Delivery Network) A mechanism that treats inference artifacts (e.g., LLM KV caches) as reusable and distributable objects, enabling cooperative acceleration across agents or gateways.

APA (Adaptive Protocol Adapter) An automated protocol adaptation function that discovers/infers external interface schemas and normalizes heterogeneous protocols (e.g., HTTP, gRPC, MQTT) into an internal standardized format.

MCP (Model Context Protocol) A reference standard for connecting AI assistants/agents to tools and data sources, used here as an example of a normalized internal interaction format.

A2A Agent-to-agent messaging format/protocol used inside the ISD. This draft treats A2A as a generic class of agent messaging and illustrates how Agent-GW routes and adapts it.

Peer Agent-GW Another Agent-GW instance in a different node/site/domain. Peer Agent-GWs may synchronize selected state or inference artifacts subject to policy.

4. Deployment Model and Trust Boundary

Many deployments distinguish "internal" versus "external" entities by a network boundary aligned with a LAN. In this draft, the Internal Semantic Domain (ISD) refers to the internal LAN/on-prem cluster where agents and enterprise tools operate under shared governance, while the External Heterogeneous Ecosystem (EHE) refers to networks and services outside that boundary.

Agent-GW logically sits at the intersection of these domains. It provides (1) semantic routing and state functions within the ISD, and (2) border adaptation functions for controlled egress/ingress across the trust boundary. Policies MAY restrict what context, memory, or inference artifacts can cross from ISD to EHE.

The internal/external distinction is a deployment choice. The same Agent-GW architecture can be deployed as a pure internal hub (no external egress), a border gateway, or a hybrid topology with peer synchronization.

5. Network and Infrastructure Requirements

Agent interactions are typically context-heavy, short-lived, and driven by high-level goals. To support this, the infrastructure SHOULD satisfy the following requirements:

***Intent-Based Addressing:** The infrastructure SHOULD support addressing based on capabilities and intent rather than topology.

***Stateful Context Management:** Agentic workflows often involve multi-step reasoning where context accumulates. The gateway MUST support policy-controlled state retention and retrieval.

***Heterogeneous Interoperability:** The ecosystem includes diverse protocols. The gateway SHOULD provide automated adaptation layers (e.g., APA) and normalization into standardized internal formats.

***Dynamic Capability Discovery:** The gateway SHOULD provide real-time capability discovery and health/status tracking for dispatch decisions.

***Inference Efficiency:** The gateway MAY cache and share inference artifacts such as KV caches (KDN) to reduce redundant computation and improve TTFT.

***Trust Boundary Enforcement:** The gateway MUST enforce policies for data privacy, context leakage prevention, and capability spoofing mitigation, especially across ISD/EHE boundaries.

6. Architecture Overview

This section describes the reference architecture of the Agent Communication Gateway (Agent-GW). Agent-GW functions as a semantic intermediary operating at the application and cognitive layers, with explicit separation between core semantic/state functions and border adaptation functions.

6.1. Internal/External Entity Relationship

Figure 1 illustrates Agent-GW within a LAN-scoped Internal Semantic Domain (ISD) and its controlled interfaces to an External Heterogeneous Ecosystem (EHE). This figure is intentionally structured to highlight (a) internal agents and clients, (b) Agent-GW core state/routing functions, (c) border adaptation functions, and (d) southbound targets including legacy APIs, native agents, and peer gateways.

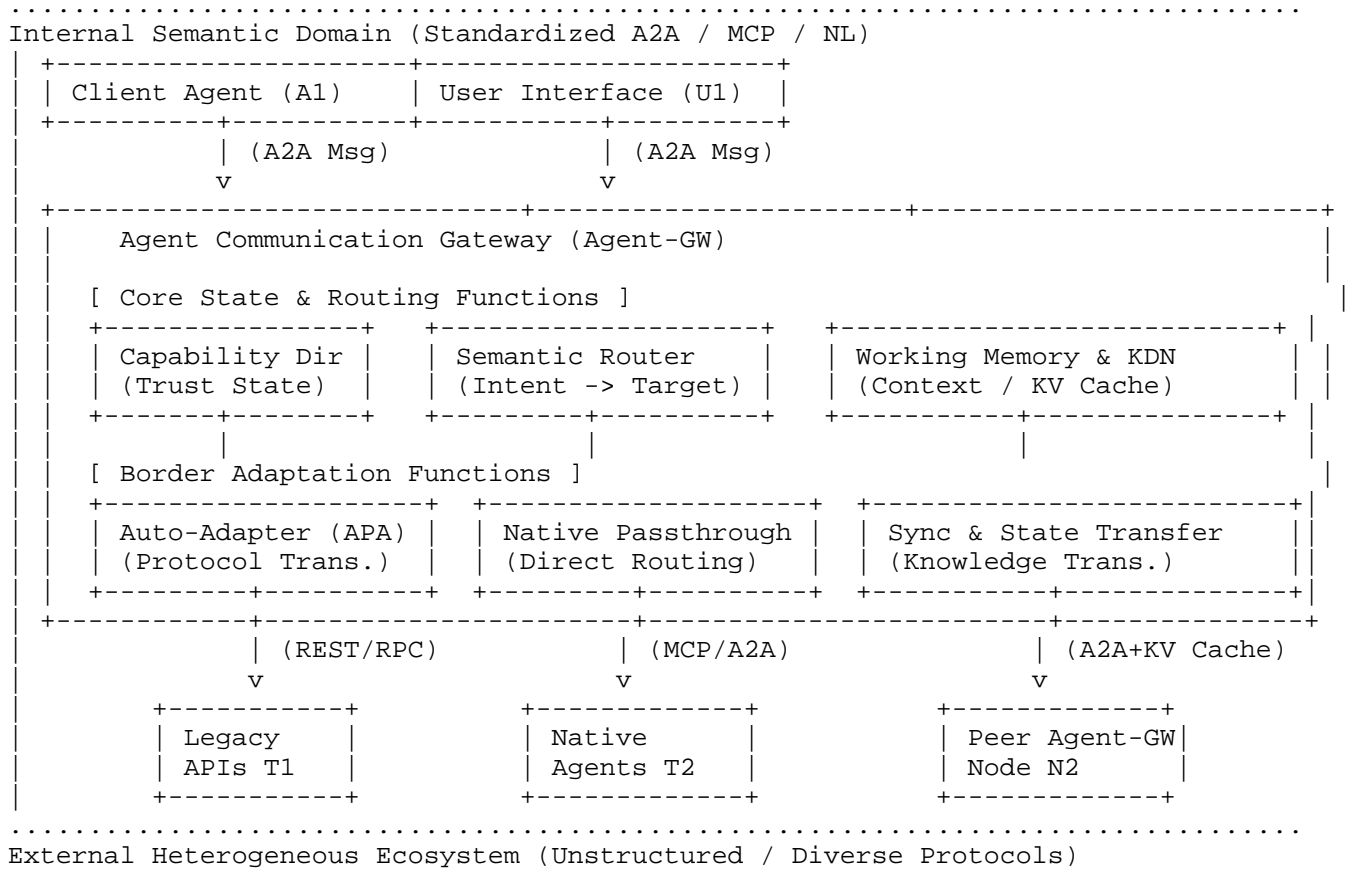


Figure 1: Agent-GW Entity Relationship across Internal Semantic Domain and External Ecosystem

Operationally, messages originating from internal clients/agents enter Agent-GW via standardized internal formats (e.g., A2A or MCP). If a target resides in the EHE, Agent-GW invokes APA for protocol adaptation and applies egress policies to prevent unintended context leakage.

6.2. Functional Planes

Agent-GW can be described as four logical planes:

(1) Ingress/Access: protocol detection, authentication, sandboxing, normalization. (2) Cognitive Orchestration: intent parsing, planning, semantic routing, dispatch, observability hooks. (3) Knowledge & State: working memory, experience/evolutionary memory, KDN cache and artifact management. (4) Egress/Ecosystem Interface: drivers for legacy systems, native agents, physical-world bridges, and peer sync.

7. Protocol Model and Input/Output Examples

Agent-GW supports a mixed protocol environment. Within the ISD, interactions are RECOMMENDED to use standardized agent messaging (e.g., A2A or MCP). For southbound access to targets, Agent-GW MAY translate into external protocols such as REST/HTTP, gRPC, MQTT, OPC UA, ROS, or vendor-specific SDKs.

The following subsections provide illustrative (non-normative) examples of message shapes and I/O mapping. These examples are intended to clarify how semantic routing, working memory, and adaptation interact.

7.1. Example: A2A Request Normalization

Illustrative A2A message (ingress) that Agent-GW normalizes into an internal semantic request:

```
{
  "a2a_version": "1",
  "session_id": "s-123",
  "from": "agent:A1",
  "intent": "Inspect Assembly Line B",
  "constraints": {
    "latency_ms": 800,
    "privacy": "internal_only"
  },
  "context_ref": ["wm://s-123/ctx"]
}
```

Illustrative normalized semantic request inside Agent-GW (after parsing and policy checks):

```
{
  "session_id": "s-123",
  "intent": {
    "task": "inspect",
    "target": "assembly_line",
    "id": "B"
  },
  "routing_hints": {
    "privacy_scope": "ISD",
    "required_capabilities": ["iot_read", "robot_navigation"]
  },
  "context": {
    "working_memory_keys": ["ctx", "last_actions"],
    "kdn_cache_allowed": true
  }
}
```

7.2. Example: Southbound Output Mapping

When dispatching to a legacy IoT array, Agent-GW MAY translate a sub-task into MQTT or REST. When dispatching to an embodied agent, Agent-GW MAY translate into a ROS bridge. These mappings are policy-controlled and can be produced by APA or pre-registered drivers.

Example REST payload for a legacy API target:

```
{
  "req": "temp_read",
  "loc": "Line B"
}
```

Example ROS command topic for an embodied agent target:

```
/cmd_vel
/navigate_to
```

8. Infrastructure Functions

8.1. Agent Identification and Capability Directory

This function establishes a root of trust for the agent network and mitigates capability spoofing. Agent-GW maintains a dynamic directory where entries represent active, verified agent states rather than static records.

***Cryptographic Identity:** Participating agents SHOULD possess a cryptographic Agent ID (AID) bound to credentials (e.g., X.509 certificate). An agent registers by submitting an AgentCard that binds identity to a capability descriptor (e.g., capability hash, policy tags).

***Capability Claim and Verification (CCV):** To reduce malicious registration, Agent-GW MAY implement challenge-response verification based on metamorphic testing principles (semantic variants of a task) to evaluate functional consistency without requiring access to internal model weights.

***Semantic Heartbeat:** To maintain freshness, Agent-GW MAY periodically verify Layer-7 functional integrity (beyond L3 keep-alives). Agents failing challenges MAY be dynamically quarantined or pruned.

8.2. Automated Protocol Adaptation and Interface Normalization (APA)

Residing at the border adaptation functions, APA normalizes heterogeneous external protocols (HTTP, MQTT, gRPC, proprietary SDKs) into an internal standardized request format (e.g., MCP or A2A). For poorly documented interfaces, APA MAY apply active probing to infer schemas and refine bindings with feedback loops.

8.3. Infrastructure-Level Agent Evaluation and Compliance

Agents are often black boxes. Agent-GW introduces infrastructure-level evaluation to estimate reliability and compliance without access to model weights. Using oracle-free metamorphic testing, Agent-GW generates semantic variants of tasks and evaluates response consistency. Results MAY contribute to a dynamic reliability score used in routing.

8.4. Dynamic Orchestration and Semantic Routing

Static routing tables are insufficient for dynamic collaboration. Agent-GW performs semantic routing by decomposing complex intents into a DAG of sub-tasks and dispatching them to suitable targets based on capability matching, trust score, privacy constraints, and operational metrics.

8.5. Evolutionary Knowledge Management

Agent-GW MAY incorporate evolutionary memory that captures execution traces, success/failure outcomes, and user corrections. This enables continuous improvement in routing policies and can provide feedback guidance to terminal agents.

8.6. Collaborative Inference Acceleration (KDN)

Multi-agent workflows often repeat reasoning over shared context. KDN treats inference artifacts (e.g., LLM KV caches) as reusable objects, enabling sharing across co-located agents or peer Agent-GWs subject to policy. This can reduce TTFT and total compute.

9. Representative Deployment Scenarios

This section provides representative scenarios with explicit internal/external boundaries, and concrete input/output protocol examples. These scenarios are illustrative and non-normative.

9.1. Scenario 1: Enterprise Copilot with Local Secure Routing and External Egress

An employee copilot receives a natural language request that requires both private on-prem data and public market information. Agent-GW routes sensitive processing to an internal SLM/LLM while allowing limited external API egress for public data, enforcing privacy and context minimization.

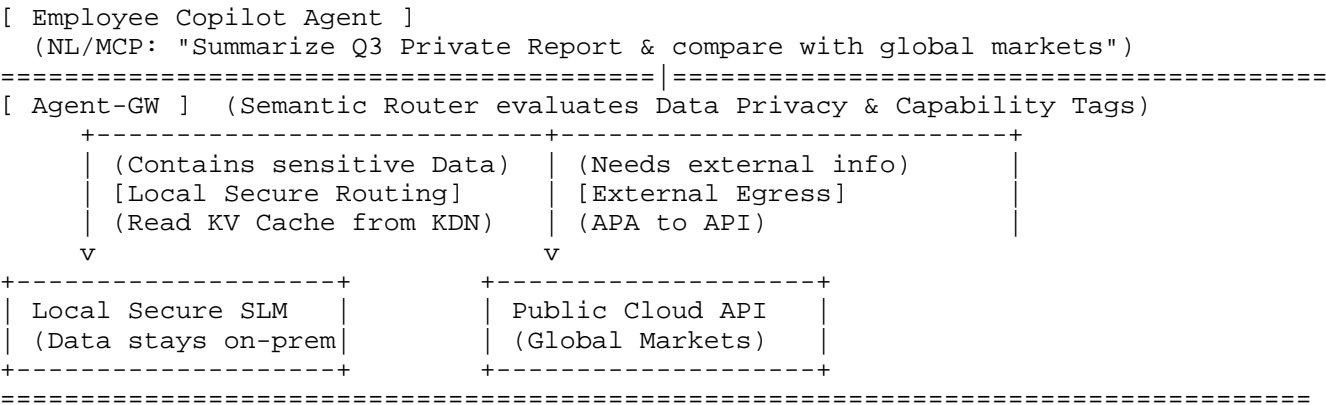


Figure 2: Enterprise Copilot: Split Routing by Privacy and Capability Tags

Example ingress request (MCP-like) and split dispatch:

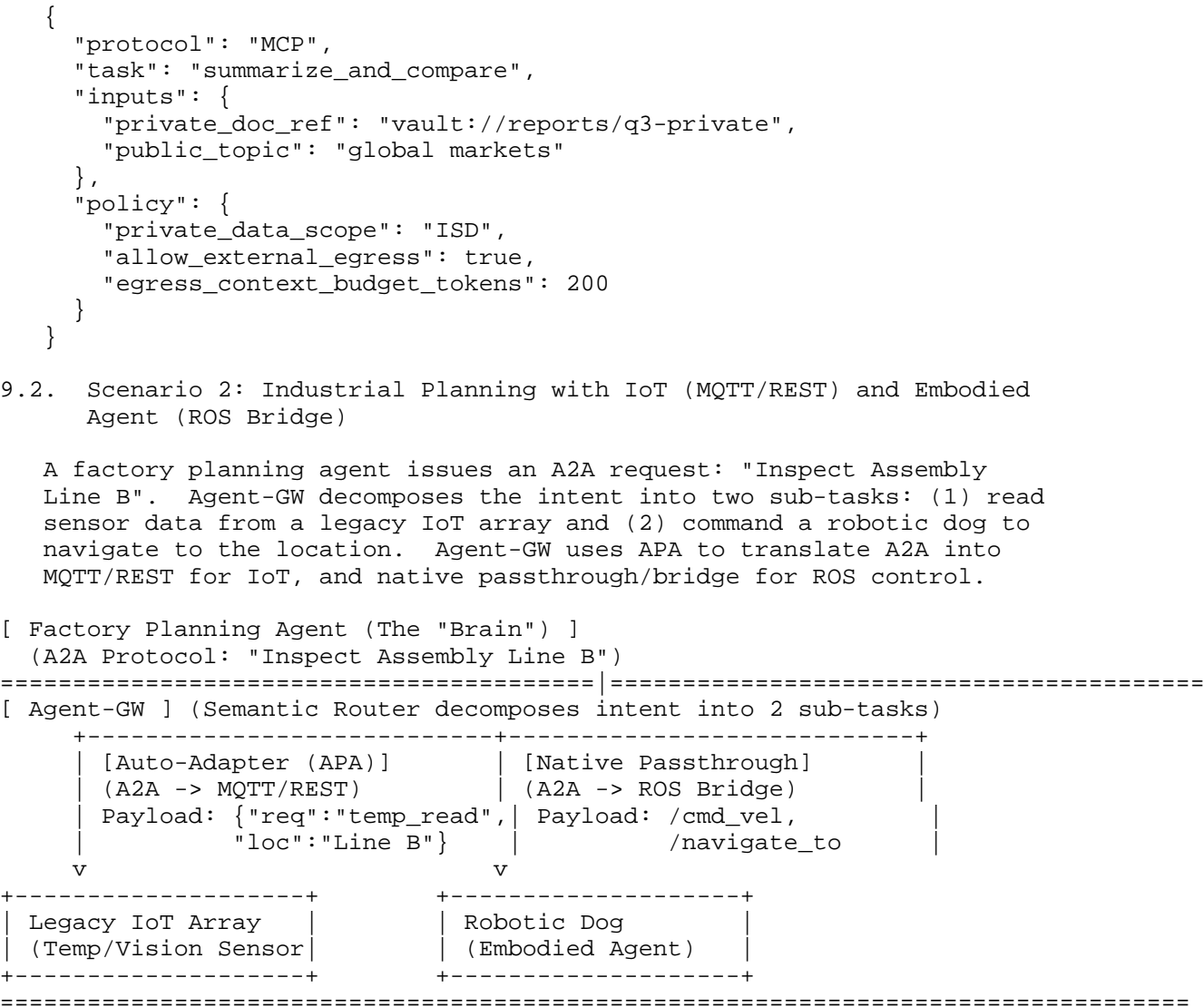


Figure 3: Industrial Scenario: Intent Decomposition and Protocol Translation

Illustrative sub-task outputs:

```

{
  "subtasks": [
    {
      "id": "t1",
      "target": "LegacyIoTArray",
      "protocol": "MQTT/REST",
      "payload": {"req": "temp_read", "loc": "Line B"}
    },
    {
      "id": "t2",
      "target": "RoboticDog",
      "protocol": "ROS",
      "payload": {"topic": "/navigate_to", "args": {"loc": "Line B"}}
    }
  ]
}

```

9.3. Scenario 3: Peer Agent-GW Synchronization and Inference Artifact Transfer

For multi-site deployments, a local Agent-GW MAY synchronize selected working memory snapshots or KDN artifacts with a peer Agent-GW. This supports mobility, disaster recovery, and cooperative acceleration. Synchronization MUST be policy-gated and can be limited to anonymized summaries or encrypted artifacts.

Example: transfer a session context digest and a KV cache handle rather than full raw prompts.

```

{
  "sync": {
    "peer": "agent-gw://node-n2",
    "session_id": "s-123",
    "transfer": {
      "working_memory_digest": "sha256:...",
      "kdn_artifact_handle": "kdn://artifact/kv/abc",
      "encryption": "HPKE",
      "policy_tags": ["no_raw_pii", "ttl_10m"]
    }
  }
}

```

10. Security Considerations

Introducing an active Agent-GW raises specific security challenges including agent identity spoofing, capability poisoning, context leakage, inference artifact theft, and cross-boundary data exfiltration.

Agent-GW deployments MUST define explicit trust boundaries (e.g., ISD vs EHE) and enforce policies for: (1) authentication/authorization for agent registration and dispatch, (2) privacy scoping for working memory, (3) egress filtering and context minimization, (4) encryption and access control for KDN artifacts, (5) observability and audit trails for routing decisions and protocol adaptation.

11. IANA Considerations

This document has no IANA actions at this time.

12. Acknowledgements

TBD

13. References

13.1. 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/info/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/info/rfc8174>>.

Authors' Addresses

Xiaohui Xie
Tsinghua University
Email: xiexiaohui@tsinghua.edu.cn

Zian Wang
Beijing University of Posts and Telecommunications
Email: zianwang@bupt.edu.cn

Tianshuo Hu
Tsinghua University
Email: huts22@mails.tsinghua.edu.cn

Yong Cui
Tsinghua University
Email: cuiyong@tsinghua.edu.cn