Independent Submission
Internet-Draft
Intended status: Informational
Expires: 24 September 2026

C. Hood
Nomotic, Inc.

23 March 2026

Agent Transfer Protocol (AGTP)
draft-hood-independent-agtp-02

Abstract

AI agents and agentic systems generate a growing volume of intent-driven, unstructured, and undifferentiated traffic that flows through HTTP indistinguishably from human-initiated requests. HTTP lacks the semantic vocabulary, observability primitives, and identity mechanisms required by agent systems operating at scale. Existing protocols described as Agent Group Messaging Protocols (AGMP), including MCP, ACP, A2A, and ANP, are messaging-layer constructs that presuppose HTTP as their transport. They do not address the underlying transport problem.

This document defines the Agent Transfer Protocol (AGTP): a dedicated application-layer protocol for AI agent traffic. AGTP provides agent-native intent methods (QUERY, SUMMARIZE, BOOK, SCHEDULE, LEARN, DELEGATE, COLLABORATE, CONFIRM, ESCALATE, NOTIFY, DESCRIBE, SUSPEND), protocol-level agent identity and authority headers, and a status code vocabulary designed for the conditions AI agent systems encounter. AGTP SHOULD prefer QUIC for new implementations and MUST support TCP/TLS for compatibility and fallback. It is designed to be composable with existing agent frameworks, not to replace them. Version 02 introduces capability discovery (DESCRIBE), resource budget signaling and enforcement, optional RATS-aligned execution attestation, observability hooks, network zone isolation, session suspension as a method, and normative composition profiles with AGMP (Agent Group Messaging Protocols). Version 02 enables dynamic capability negotiation and resource-aware governance.

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/.

Copyright Notice

Table of Contents

1. Introduction

Note Regarding Intellectual Property: Implementers should be aware that extensions and certain mechanisms referenced in this document -- including the Agent Certificate extension (Section 7.2), the ACTIVATE method, the Agent Birth Certificate mechanism (Section 5.7), and the .agent and .nomo file format specifications (Section 2) -- may be subject to pending patent applications by the author. The core AGTP specification is intended for open implementation without royalty obligation. The licensor is prepared to grant a royalty-free license to implementers consistent with [RFC8179]. IPR disclosures: https://datatracker.ietf.org/ipr/ -- see also Section 7.7.

1.1. Background

The deployment of AI agents and multi-agent systems is accelerating across enterprise, research, and consumer contexts. These systems execute complex, multi-step workflows, querying data sources, booking resources, delegating subtasks to peer agents, and escalating decisions to human principals, with minimal or no human supervision per transaction.

Unlike human-initiated web traffic, agent-generated traffic is dynamic, high-frequency, intent-driven, and often stateful across sequences of related requests. The infrastructure carrying this traffic was not designed with these properties in mind.

1.2. Limitations of HTTP for Agent Traffic

HTTP has served as the internet's primary application-layer transport for over three decades. Its evolution through HTTP/2 [RFC7540] and HTTP/3 [RFC9114] has improved performance, multiplexing, and latency. However, the fundamental model of HTTP being stateless, resource-oriented, human-initiated request/response, creates specific failures when applied to agentic systems at scale:

* Traffic indistinguishability: Agent-generated requests are structurally identical to human-initiated requests at the transport layer. Operators cannot identify, route, or govern agent traffic without application-layer instrumentation.

* Method vocabulary mismatch: HTTP's method set (GET, POST, PUT, DELETE, PATCH) describes resource operations. Agent traffic expresses purposeful intent, summarize, book, delegate, escalate. The mismatch forces intent into request bodies, invisible to protocol-level handlers.

* Identity and attribution absence: HTTP carries no native mechanism for asserting agent identity, declared authority scope, or the principal accountable for an agent's actions.

* Session semantics mismatch: HTTP's stateless model is optimized for isolated request/response cycles. Agent workflows are inherently stateful sequences.

1.3. Why Not Evolve HTTP?

A natural question is whether these limitations could be addressed by extending HTTP rather than defining a new protocol. There are three specific reasons why HTTP extension is not the preferred path.

First, the HTTP method registry is effectively frozen for new semantics. [RFC9110] defines the HTTP method registry with IETF Review as the registration procedure, meaning new methods require a full IETF consensus process and must be backward-compatible with existing HTTP implementations. Adding intent-based verbs (SUMMARIZE, DELEGATE, ESCALATE) to HTTP would require every HTTP client, server, proxy, and middleware component to ignore or handle unknown methods gracefully, a compatibility constraint that limits how agent-specific semantics can be expressed at the protocol level.

Second, HTTP carries decades of backward-compatibility constraints. Features such as persistent agent identity headers, authority scope declarations, and session-level governance semantics would require HTTP extensions that interact unpredictably with existing caching, proxy, and CDN behavior designed for human-generated traffic patterns.

Third, the observability goal making agent traffic distinguishable from human traffic at the infrastructure layer cannot be achieved by adding fields to HTTP. Infrastructure components route and filter HTTP traffic based on methods and headers that are identical across agent and human requests. A protocol-level separation is necessary to give infrastructure the signal it needs.

AGTP is therefore designed as a dedicated protocol rather than an HTTP extension. HTTP and AGTP coexist: human traffic continues to flow over HTTP; agent traffic flows over AGTP. The two protocols serve different classes of network participant.

Note: The abbreviation AGTP is used in this document to distinguish the Agent Transfer Protocol from the Authenticated Transfer Protocol (ATP) working group currently chartered within the IETF. The URI agtp:// is proposed for IANA registration as a new and distinct scheme.

1.4. Motivation for a Dedicated Protocol

These limitations are architectural, not implementational. They cannot be resolved by better middleware or application code layered on HTTP. They require a protocol designed from first principles for AI agent systems.

AGTP is that protocol. It provides a dedicated transport environment for agent traffic with: native intent-based methods, mandatory agent identity headers, protocol-level authority scope declaration, and a status code vocabulary for the conditions AI systems encounter.

1.5. Scope and Target Audience

This document covers AGTP architecture, design principles, stack position, request and response header format, agent-native method definitions and semantics, status code vocabulary, security considerations, and IANA considerations.

The Agent Certificate extension for cryptographic binding of agent identity to AGTP header fields is described at a high level in Section 7.2. Full specification is provided in a separate companion document: [AGTP-CERT]. That extension may be subject to pending intellectual property claims; see Section 7.7 and the IPR Notice preceding the Abstract.

Target audience: AI agent developers, protocol designers, cloud and network infrastructure providers, enterprise security and compliance architects, and standards community participants.

1.6. AGTP as the Transport Foundation for Agent Group Messaging Protocols

AGTP is the purpose-built transport and governance layer for Agent Group Messaging Protocols (AGMPs): the category of higher-layer AI agent messaging standards that includes the Model Context Protocol (MCP) [MCP], the Agent-to-Agent Protocol (A2A) [A2A], the Agent Communication Protocol (ACP) [ACP], and emerging others.

AGMPs define what agents say. AGTP defines how those messages move, who sent them, and under what authority. AGTP provides the narrow-waist foundation that AGMPs inherit without modification: intent-native methods, mandatory agent identity and scoping, resource budget enforcement, observability hooks, and normative composition profiles. A deployment running any AGMP over AGTP gains transport-level governance without changes to the messaging layer.

The AGMP category term is introduced in this document to provide a stable collective reference for the class of protocols that AGTP serves as substrate. It is not a formal IETF term of art; it is a descriptive classification. Individual AGMP specifications retain their own names and development paths. AGTP does not govern, modify, or supersede any AGMP.

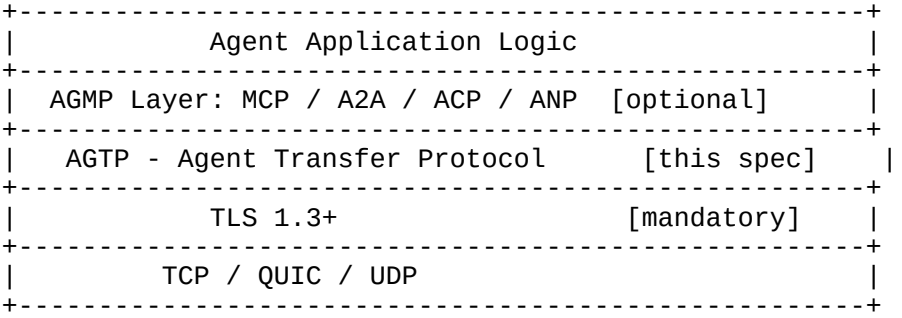

Figure 1: AGTP as Substrate for AGMPs

2. Terminology

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.

Agent: An AI software system that executes tasks, makes decisions, and takes actions without continuous human supervision per transaction.

Principal: The human, organization, or system that authorized an agent to act and is accountable for its actions.

Agent-ID: A unique identifier for a specific agent instance, present in all AGTP request headers.

Principal-ID: The identifier of the principal on whose behalf an agent operates.

Authority-Scope: A declared set of permissions defining what actions an agent is authorized to take, in the format domain:action or domain:*.

Intent Method: An AGTP method name expressing the agent’s purpose, as distinguished from HTTP resource-operation verbs.

Delegation Chain: An ordered record of Agent-IDs representing the sequence of delegations that produced the current request.

Escalation: An agent’s intentional deferral of a decision or action to a human principal or higher-authority agent.

Attribution Record: A logged record of an agent action sufficient for audit and compliance purposes.

Session: An AGTP persistent connection context shared across multiple method invocations within a single agent workflow.

SEP (Scope-Enforcement Point): An AGTP-aware infrastructure component, load balancer, gateway, and proxy, that enforces Authority-Scope compliance without application-layer access. Requires the Agent Certificate extension ([AGTP-CERT]).

Agent Package (.agent): A portable, open deployment artifact for an

AI agent. An .agent file contains an embedded Agent Manifest, an integrity hash covering all package contents, and a behavioral trust score computed at packaging time. The .agent format is an open specification. It is analogous to a container image: a self-describing, portable unit of deployment. The .agent suffix is a file format designator and *MUST NOT* appear as a hostname component or top-level label in agtp:// URIs. Note: the .agent file format specification may be subject to pending patent claims by the author; see Section 7.7.

Governed Agent Package (.nomo): A deployment artifact in the .nomo format, which extends the .agent format with a CA-signed certificate chain binding the package to a verified governance zone and issuing principal. The .nomo format is to .agent as HTTPS is to HTTP: the same structural foundation with an added layer of cryptographic trust. A .nomo package is required for agents operating at Trust Tier 1 (see Section 5.2). The .nomo suffix is a file format designator and *MUST NOT* appear as a hostname component in agtp:// URIs.

The name derives from the Greek _nomos_ ($\nu \mu o$), meaning law, rule, or governance, the same root that underlies _autonomy_ (self-law), _nomocracy_ (rule of law), and _onomastics_. A .nomo package is literally an agent operating under law: its behavior is bounded by a cryptographically enforced governance context at the packaging layer. Note: the .nomo file format specification may be subject to pending patent claims by the author; see Section 7.7.

Agent Transfer Document (.agtp): The wire-level manifest document format defined by this specification. An .agtp document is a signed JSON structure containing the fields defined in Section 5.5 (Agent Manifest Document). It is the output format returned by all AGTP URI resolution requests. Both .agent and .nomo packages produce .agtp documents when queried; the .agtp format is the protocol's canonical representation of agent identity and is independent of the underlying packaging format. The .agtp suffix *MAY* appear in filenames for stored manifest documents but *MUST NOT* appear in agtp:// URIs. The Content-Type for .agtp documents is application/agtp+json.

URI (AGTP): An agtp:// scheme URI that identifies an agent or agent namespace. AGTP URIs are addresses, not filenames. File extensions (.agent, .nomo, .agtp) *MUST NOT* appear in canonical AGTP URIs. See Section 5.1 for the canonical URI forms and resolution semantics.

Agent Namespace Document: A cryptographically signed application/

agtp+json document returned in response to a request targeting an organization's agent registry root (e.g., agtp://acme.tld/agents). Lists all Active agents registered under the organization's governance zone. The document is generated and re-signed by the governance platform on any registry change. It is not a manually editable file. See Section 5.4.

Agent Manifest Document: A cryptographically signed application/agtp+json document returned in response to a request targeting a specific agent (e.g., agtp://acme.tld/agents/customer-service). Contains the agent's birth certificate fields, lifecycle state, behavioral trust score, authority scope categories, supported methods, and governance zone. Derived directly from the agent's .agent or .nomo package; the package integrity hash is verified before the manifest is served. See Section 5.5.

Agent Birth Certificate: A cryptographically signed identity document issued to an agent at registration time by a governance platform. The Birth Certificate is the genesis record of an agent's existence: it establishes the agent's identity, ownership, authorized scope, behavioral archetype, and governance zone before the agent takes any action. Authority is issued through the Birth Certificate; it is never self-assumed.

The Birth Certificate is the source document from which the Agent Manifest Document (Section 5.5) is derived when an AGTP URI is resolved. The certificate_hash field of the Birth Certificate is the basis for the agent's canonical Agent-ID. In this sense the Birth Certificate functions as the agent's social security number: issued once at creation, permanently bound to the individual, and the authoritative identity record from which all other identity representations derive.

Birth Certificate fields map directly to AGTP protocol headers: agent_id maps to the Agent-ID header; owner maps to the Principal-ID header; scope maps to the Authority-Scope header. See Section 5.7.

Anonymous agents are ungovernable. Without a Birth Certificate, there is no mechanism to trace decisions to a responsible principal, enforce scope boundaries, or maintain a meaningful audit trail. Note: the Agent Birth Certificate mechanism may be subject to pending patent claims by the author; see Section 7.7.

Governance Token: A signed, time-limited JWT artifact issued by a governance runtime that encodes a specific governance decision for a specific action. Governance tokens are the runtime companion to the static Birth Certificate: where the Birth Certificate

establishes persistent identity, the Governance Token carries a bounded authorization for a single action or session. Tokens carry the governance verdict (ALLOW, DENY), the agent ID, action details, trust score dimensions, issuer identity, and expiry. Default TTL: 30 seconds. Tokens *MUST NOT* be reused across actions; each action requires a fresh evaluation and a fresh token.

Trust Tier: A classification assigned to an agent based on the strength of identity verification backing its registration. Tier 1 (Verified): org anchor is a real DNS domain with confirmed ownership and a .nomo governed package. Tier 2 (Org-Asserted): org label is present but DNS ownership is unverified; .agent package acceptable. Tier 3 (Experimental): X- prefix required; not discoverable through the public AGTP registry. See Section 5.2.

AGMP (Agent Group Messaging Protocol): The collective term for higher-layer AI agent messaging standards that operate over AGTP as their transport substrate, including MCP [MCP], A2A [A2A], ACP [ACP], and ANP [ANP]. AGMPs define what agents say to each other. AGTP defines how those messages move. The term is introduced in this document as a descriptive classification; it is not a formal IETF term of art.

DESCRIBE: An AGTP Tier 1 core method that returns the declared capabilities, supported modalities, method vocabulary, and versioned feature set of a specific agent endpoint. Distinguished from URI resolution (which returns identity) by returning operational capability metadata suitable for pre-task negotiation. If the capability_domains parameter is omitted, the server *SHOULD* return all supported domains. Category: ACQUIRE.

SUSPEND (method): An AGTP Tier 1 core method that places a specific active session workflow into a recoverable paused state, issuing a resumption nonce for re-entry. Distinguished from the lifecycle SUSPEND event (Section 6.7.6): method-level SUSPEND is session-scoped and does not affect the agent’s registry lifecycle state or Birth Certificate validity. Category: ORCHESTRATE.

Budget-Limit: A request header declaring the maximum resource consumption the principal authorizes for a method invocation, expressed as space-separated unit:value tokens drawn from the IANA AGTP Budget Unit Registry. Example: Budget-Limit: tokens=5000 compute-seconds=120 financial=10.00USD ttl=3600. Exceeding the declared limit *MUST* cause the server to return 452 Budget Exceeded rather than continue execution. Note: ttl= is RECOMMENDED to bound budget lifetime.

AGTP-Zone-ID: A request header declaring the network zone or organizational boundary within which a request must be processed. Scope-Enforcement Points (SEPs) *MUST* enforce zone boundaries and *MUST* return 453 Zone Violation if a DELEGATE or COLLABORATE request would route outside the declared zone.

3. Problem Statement

AGTP is motivated by three distinct, compounding failures in how current internet infrastructure handles AI agent traffic.

3.1. Problem 1: Undifferentiated Agent Traffic on HTTP

AI agents generate intent-driven, structured traffic that is functionally invisible to the infrastructure it traverses. This traffic flows through HTTP alongside human traffic with no protocol-level differentiation. Observability failure, routing inefficiency, and security blindness result, operators cannot determine what fraction of traffic is agent-generated without application-layer instrumentation that is expensive, inconsistent, and easy to circumvent.

AGTP response: a dedicated protocol environment for agent traffic. Infrastructure can distinguish, route, monitor, and govern agent traffic natively.

3.2. Problem 2: Semantic Mismatch Between Agent Intent and Available Methods

AI agents operate on intent. HTTP’s method vocabulary was designed to describe operations on resources, not purposeful action. When an agent intends to SUMMARIZE a document, BOOK a resource, and SCHEDULE a sequence, all three arrive as POST requests. The server receives identical verbs with meaningfully different intent buried in request bodies, invisible to any protocol-level handler.

AGTP response: a vocabulary of agent-native methods that express intent at the protocol level.

3.3. Problem 3: No Protocol-Level Identity, Authority, or Attribution for Agents

When an AI agent takes an action, there is currently no protocol-level mechanism to verify who authorized this agent, what scope of authority it holds, which principal is accountable for its actions, or whether it is the agent it claims to be. Accountability gaps, authority laundering, auditability failure, and multi-agent trust collapse result.

AGTP response: agent identity and authority scope embedded in protocol headers on every request, with an optional Agent Certificate extension for cryptographic verification.

3.4. Problem Summary

#	Problem	Current Failure	AGTP Response
1	Undifferentiated traffic	HTTP cannot separate agent traffic	Dedicated protocol environment
2	Semantic mismatch	HTTP verbs obscure agent intent	Native intent-based method vocabulary
3	No protocol-level identity	Attribution is untraceable	Agent identity and scope in headers

Table 1: Summary of Problems Addressed by AGTP

4. Related Work and Existing Approaches

4.1. HTTP/REST as the De Facto Standard

HTTP remains the universal transport for all agent traffic currently deployed. REST conventions layered on HTTP provide a degree of semantic structure, but REST remains a resource-manipulation paradigm. As described in Section 1.3, evolving HTTP to address agent-specific needs is constrained by the frozen method registry, backward-compatibility requirements, and the impossibility of achieving infrastructure-level traffic differentiation through HTTP extensions alone.

4.2. Existing Agent Group Messaging Protocols

MCP [MCP] (Model Context Protocol, Anthropic): Defines structured communication between AI models and tools/resources. Runs over HTTP. Addresses tool-calling semantics, not agent traffic transport.

ACP [ACP] (Agent Communication Protocol, IBM): Defines messaging semantics for agent-to-agent communication. Runs over HTTP.

A2A [A2A] (Agent-to-Agent Protocol, Linux Foundation): Defines inter-agent communication and task delegation semantics. Runs over HTTP.

ANP [ANP] (Agent Network Protocol): Defines discovery and communication for networked agents. Runs over HTTP.

All of these are messaging protocols. They define what agents say to each other. They do not define how agent traffic moves across a network. Each presupposes HTTP as its transport and inherits all of HTTP's limitations for agentic systems.

4.3. Transport-Layer Alternatives

gRPC: High-performance RPC over HTTP/2. Strong typing and efficient serialization. Does not address agent-specific semantics, identity, or authority.

WebSockets: Persistent bidirectional connections over HTTP. Useful for real-time communication but does not address method semantics or identity.

QUIC [RFC9000]: Modern multiplexed transport with reduced connection overhead. AGTP *SHOULD* prefer QUIC for new implementations. QUIC is a transport primitive; AGTP is the application-layer protocol above it.

4.4. The Critical Distinction: Messaging vs. Transport

The most important positioning principle for AGTP is the distinction between messaging protocols and transport protocols. MCP, ACP, A2A, and ANP are messaging protocols, they define what agents say. AGTP defines how agent traffic moves.

An analogy: SMTP is a messaging protocol that runs over TCP. SMTP does not replace TCP. Saying "TCP is unnecessary because SMTP exists" is a category error. The same logic applies here. MCP and its peers define agent messaging semantics. AGTP defines the transport environment those messages move through.

4.5. AGTP Positioning: The Proposed Stack

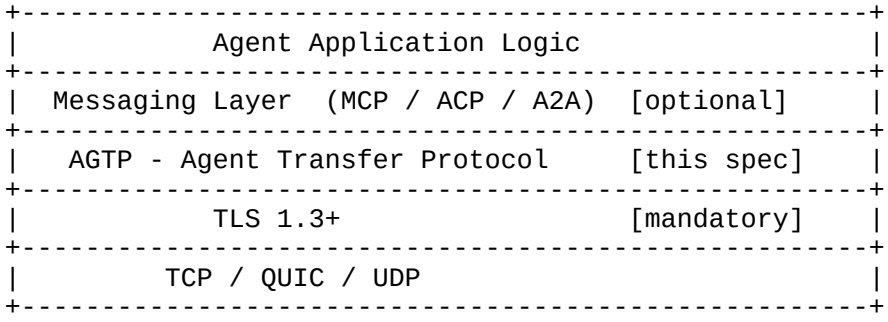

Figure 2: AGTP in the Protocol Stack

AGTP is not a replacement for messaging protocols. Agents using MCP or A2A route those messages over AGTP and gain transport-level observability and identity without modifying the messaging layer. AGTP-native agents that do not use a separate messaging protocol interact with AGTP methods directly.

5. Protocol Overview

5.1. Stack Position

AGTP is an application-layer protocol. It operates above the transport layer (TCP, UDP, or QUIC) and is wrapped by TLS. It sits below any agent messaging protocol in deployments that use one.

* *SHOULD* prefer QUIC [RFC9000] [RFC9001] for new deployments (lower latency, multiplexing without head-of-line blocking, 0-RTT connection establishment).

* *MUST* support TCP/TLS as a fallback for compatibility with existing infrastructure.

* *MAY* run over UDP where QUIC is not available, subject to implementor-defined reliability guarantees.

Suggested port assignment (subject to IANA assignment. See Section 8):

* AGTP/QUIC: port 8443 (proposed)

* AGTP/TCP+TLS: port 8080 (proposed)

5.2. Design Principles

Minimalist core: The base spec defines only what is necessary for

agent traffic differentiation, method semantics, and identity headers. Extensions belong in companion specifications.

Extensible by design: New methods are registered through an IANA-managed Method Registry. New header fields follow a defined extension convention. Additive changes do not require a version increment.

Agent-native: Every design decision assumes the initiating party is an AI system, not a human.

Secure by default: TLS 1.3 or higher is mandatory. Unencrypted AGTP connections *MUST* be rejected. Agent identity headers are present on every request.

Observable by design: Native metadata in every AGTP header provides the minimum information needed for routing, monitoring, and audit without application-layer instrumentation.

Composable: AGTP works alongside existing agent messaging protocols without requiring modification to those protocols.

5.3. Connection Model

AGTP uses a persistent session model by default, reflecting the reality that agents typically execute multi-step workflows rather than isolated single requests. An AGTP session is established with a single TLS handshake including agent identity assertion, persists across multiple method exchanges, carries a Session-ID header identifying the agent's task context, and terminates on explicit session close or inactivity timeout (RECOMMENDED minimum: 60 seconds).

Per-request (stateless) mode is supported for constrained environments. In stateless mode, agent identity headers *MUST* be present on every individual request.

5.4. Header Format

5.4.1. Request Headers

Field	Required	Description
AGTP-Version	*MUST*	Protocol version. Current: AGTP/1.0
AGTP-Method	*MUST*	The agent intent method (see

		Section 6)
Agent-ID	*MUST*	Opaque identifier for the requesting agent instance
Principal-ID	*MUST*	Identifier of the human or system that authorized this agent
Authority-Scope	*MUST*	Declared scope of actions this agent is authorized to take
Session-ID	*SHOULD*	Identifies the current task/ workflow context
Task-ID	*SHOULD*	Unique identifier for this specific method invocation
Delegation-Chain	*MAY*	Ordered list of Agent-IDs if this request was delegated
Priority	*MAY*	Request priority hint: critical, normal, background
TTL	*MAY*	Maximum acceptable response latency in milliseconds
Budget-Limit	*MAY*	Max resource budget per invocation. Format: space-separated unit=value tokens. Units from IANA AGTP Budget Unit Registry.
AGTP-Zone-ID	*MAY*	Network zone boundary constraint. SEPs *MUST* enforce; return 453 if DELEGATE or COLLABORATE would exit declared zone.
Content-Schema	*MAY*	URI reference to JSON Schema describing the request body structure. Enables receivers to validate payload without LLM inference.
Telemetry-Export	*MAY*	OTLP endpoint URI for metric export, or inline to receive

			metrics embedded in the response Attribution-Record.

Table 2: AGTP Request Header Fields

5.4.2. Response Headers

Field	Required	Description
AGTP-Version	*MUST*	Protocol version
AGTP-Status	*MUST*	Numeric status code (see Section 5.5)
Task-ID	*MUST*	Echo of request Task-ID for correlation
Server-Agent-ID	*SHOULD*	Identity of the responding server or agent
Attribution-Record	*SHOULD*	Signed record of the action taken, for audit. *MAY* include RATS attestation evidence and inline telemetry when Telemetry-Export is set to inline.
Continuation-Token	*MAY*	Token for retrieving additional results in streaming contexts
Supported-Methods	*SHOULD* (on session open)	List of AGTP methods supported by this server
Cost-Estimate	*MAY*	Estimated resource consumption in Budget-Limit unit format. Returned by QUOTE; *MAY* appear on any response as an informational signal.
Attestation-Evidence	*MAY*	RATS attestation evidence token or reference URI per [RFC9334]. Format indicated

			by attestation_type in response body: rats-eat, rats-corim, or rats-uri.

Table 3: AGTP Response Header Fields

5.5. Status Codes

AGTP defines its own status code space. Codes 451, 452, 453, 550, and 551 are AGTP-specific with no HTTP equivalent and are registered in the IANA AGTP Status Code Registry (see Section 9.3).

Code	Name	Meaning
200	OK	Method executed successfully
202	Accepted	Method accepted; execution is asynchronous
204	No Content	Method executed; no response body
400	Bad Request	Malformed AGTP request
401	Unauthorized	Agent-ID not recognized or not authenticated
403	Forbidden	Agent lacks authority for requested action per Authority-Scope
404	Not Found	Target resource or agent not found
408	Timeout	TTL exceeded before method could execute
409	Conflict	Method conflicts with current state (e.g., BOOK on unavailable resource)
410	Gone	Agent has been Revoked or Deprecated; canonical ID is permanently retired
422	Unprocessable	Request well-formed but semantically invalid
429	Rate Limited	Agent is exceeding permitted request frequency

451	Scope Violation	Requested action is outside declared Authority-Scope. AGTP-specific
452	Budget Exceeded	Method execution would exceed the Budget-Limit declared in the request. AGTP-specific
453	Zone Violation	Request would route outside the AGTP-Zone-ID boundary. SEP-enforced. AGTP-specific
500	Server Error	Internal failure in the responding system
503	Unavailable	Responding agent or system temporarily unavailable or Suspended
550	Delegation Failure	A delegated sub-agent failed to complete the requested action. AGTP-specific
551	Authority Chain Broken	Delegation chain contains an unverifiable or broken identity link. AGTP-specific

Table 4: AGTP Status Codes

Status code 451 (Scope Violation) is a governance signal: the agent attempted an action outside its declared Authority-Scope, caught at the protocol level. Status code 452 (Budget Exceeded) is a governance signal analogous to 451: the agent's requested action is within its Authority-Scope but would consume resources beyond what the principal authorized for this invocation. Status code 453 (Zone Violation) is returned by SEPs when a DELEGATE or COLLABORATE request would route to an agent outside the declared AGTP-Zone-ID boundary. Status code 551 (Authority Chain Broken) indicates that one or more Agent-ID entries in the Delegation-Chain header cannot be verified as part of a valid delegation sequence. Status code 410 (Gone) is returned when an agent's Birth Certificate has been revoked or the agent deprecated; the canonical Agent-ID is permanently retired and *MUST NOT* be retried. All AGTP-specific status codes are operational signals, not protocol errors, and *MUST* be logged for audit purposes.

5.6. Wire Format and Content-Type

AGTP request and response bodies are encoded as JSON. The registered Content-Type for AGTP message bodies is:

Content-Type: application/agtp+json

Implementations *MUST* include this Content-Type on all AGTP requests and responses that carry a message body. Responses with no body (e.g., 204 No Content) *MUST NOT* include a Content-Type header. Binary or streaming extensions *MAY* define additional Content-Type values as part of their companion specifications.

The common structure for all AGTP request bodies:

```
{
  "method": "QUERY",
  "task_id": "task-0042",
  "session_id": "sess-a1b2c3d4",
  "parameters": { },
  "context": { }
}
```

And for all AGTP response bodies:

```
{
  "status": 200,
  "task_id": "task-0042",
  "result": { },
  "attribution": { }
}
```

5.7. Early Implementations

AGTP is a proposed specification. No production implementations exist at the time of this writing. The author encourages early prototype implementations to validate the protocol design, identify gaps, and generate feedback prior to IETF working group submission.

> If you are building an AGTP prototype or reference implementation, please share your findings via the feedback channel listed on the cover of this document. A reference implementation in Python and/or Go is planned as open-source software concurrent with or shortly after IETF I-D submission. Implementation reports are welcome and will be incorporated into subsequent draft revisions.

Implementers wishing to experiment before the formal IANA port assignment may use port 8443 (AGTP/QUIC) and port 8080 (AGTP/TCP+TLS) as working values. These values are subject to change upon final IANA assignment.

The ACTIVATE method extension, which binds .nomo governed agent packages to AGTP as a first-class activation operation, is described in a companion document and is implemented as an optional extension. Core AGTP implementations need not support ACTIVATE to be compliant with this specification.

6. Agent Identity, URI Structure, and Registration

6.1. URI Structure and Resolution Mechanics

6.1.1. Foundational Principle

AGTP URIs are addresses, not filenames. File format suffixes (.agent, .nomo, .agtp) *MUST NOT* appear in canonical agtp:// URIs. A URI resolves to an Agent Manifest Document or Agent Namespace Document derived from the underlying package; it does not expose or serve the package itself.

Implementations *MUST* treat any URI containing a file extension in the path as non-canonical and *SHOULD* issue a 301 Moved Permanently redirect to the canonical form prior to resolution.

The Canonical Agent-ID (256-bit cryptographic identifier) remains the authoritative identifier in all AGTP protocol operations. Human-readable URIs are aliases that resolve to a canonical identifier. In the event of any conflict between a human-readable URI and a canonical Agent-ID, the canonical Agent-ID *MUST* be treated as authoritative.

6.1.2. Canonical URI Forms

AGTP defines the following canonical URI forms:

Form 1. Canonical ID (cryptographic):
agtp://[256-bit-hex-id]

Form 2. Domain-anchored agent (verified identity):
agtp://[domain.tld]/agents/[agent-label]

Form 3. Subdomain-anchored agent (recommended enterprise pattern):
agtp://agtp.[domain.tld]/agents/[agent-label]

Form 4. Organization namespace root:
agtp://[domain.tld]/agents
agtp://agtp.[domain.tld]/agents

The following URI forms are explicitly invalid and *MUST* return 400 Bad Request with error code invalid-uri-form:

```
agtp://[label].agent          (.agent as hostname TLD - prohibited)
agtp://[label].nomo           (.nomo as hostname TLD - prohibited)
agtp://[domain].[label].agent (hybrid dot-notation - prohibited)
agtp://[domain].[label].nomo  (hybrid dot-notation - prohibited)
```

Note: .agent is claimed as a Web3 top-level domain by at least one blockchain naming system. URI forms that place .agent or .nomo in the hostname position are prohibited both for collision avoidance with Web3 naming systems and because they imply domain ownership without enforcing it. See Section 5.6 for Web3 guidance.

6.1.3. Non-Canonical Forms and Redirect Behavior

The following non-canonical forms *SHOULD* be redirected to their canonical equivalents. Implementations *MUST NOT* serve package contents in response to any URI form.

Received URI	Canonical Redirect Target
agtp://acme.tld/agents/ customer-service.agent	agtp://acme.tld/agents/ customer-service
agtp://acme.tld/agents/ customer-service.nomo	agtp://acme.tld/agents/ customer-service
agtp://acme.tld/agents/ customer-service.agtp	agtp://acme.tld/agents/ customer-service

Table 5: Non-Canonical URI Forms and Redirect Targets

6.1.4. Query Parameters for Format Selection

All AGTP URI resolution requests accept an optional format query parameter controlling the serialization of the returned document.

Query Parameter	Returned Representation
(none)	Agent Manifest Document, human-readable application/agtp+json
?format=manifest	Agent Manifest Document, human-readable application/agtp+json
?format=json	Agent Manifest Document, compact application/agtp+json
?format=certificate	Birth certificate fields only, application/agtp+json
?format=status	Lifecycle state and operational status only, application/agtp+json

Table 6: AGTP URI Format Query Parameters

All format variants return signed application/agtp+json content. The ?format=json parameter is intended for programmatic consumers. The default returns the full human-readable manifest suitable for browser rendering by an AGTP-aware client.

6.1.5. Resolution Mechanics

When an AGTP server receives a request targeting an agent URI, it *MUST* perform the following steps in order:

1. Parse and validate the URI. If the URI is an invalid (prohibited) form, return 400 Bad Request with error code invalid-uri-form.

2. Resolve the agent label to a canonical Agent-ID via the governance platform’s registry lookup. If no matching agent is found, return 404 Not Found.

3. Verify the registry record lifecycle state. If the agent is Suspended, return 503 Service Unavailable with lifecycle state in the response body. If the agent is Revoked or Deprecated, return 410 Gone with lifecycle state and revocation timestamp.

4. Retrieve the agent’s package (.agent or .nomo) from the package store.

5. *Verify the package integrity hash before proceeding.* If integrity verification fails, return 500 Internal Error with error code package-integrity-failure. *MUST* be logged.

6. Extract the embedded manifest from the verified package.

7. Sign the manifest document using the governance platform’s signing key. Return the signed application/agtp+json document in the format specified by the query parameter.

The package’s executable content, code, logic, and any fields not included in the manifest schema *MUST NOT* be returned at any step. URI resolution exposes identity and status exclusively.

6.1.6. Domain Anchor and Trust Tier Assignment

The org anchor in a domain-anchored URI (Form 2 or Form 3) *MUST* be validated at registration time to determine the agent’s Trust Tier.

Trust Tier 1 - Verified (DNS-anchored): The org domain is validated by DNS challenge at ACTIVATE time per [RFC8555]. The governance platform *MUST* verify that the registering party controls the DNS zone for the claimed domain before issuing a Tier 1 Birth Certificate. Requires a .nomo governed package.

Trust Tier 1 agents *MUST* have the following DNS record published and verifiable at resolution time:

_agtp.[domain.tld]. IN TXT "agtp-zone=[zone-id]; cert=[fp]"

Trust Tier 2 - Org-Asserted: The org label is present in the URI but DNS ownership has not been verified. An .agent package is acceptable. The Agent Manifest Document for Tier 2 agents *MUST* include a prominent trust_tier: 2 field and a trust_warning field with value "org-label-unverified". AGTP-aware browsers and clients *MUST* surface a visible trust indicator distinguishing Tier 2 from Tier 1.

Tier 2 agents *MUST NOT* be granted authority scopes above documents:query and knowledge:query without AGTP-CERT cryptographic identity binding.

Trust Tier 3 - Experimental: Agent label uses the X- prefix. Not

discoverable through the public AGTP registry. For development and testing only. Implementations *MUST NOT* deploy Tier 3 agents in production.

6.1.7. Subdomain Deployment Pattern

Organizations *SHOULD* deploy AGTP endpoints at a dedicated subdomain following the pattern agtp.[organization-domain.tld] (e.g., agtp.acme.tld). This is the recommended enterprise deployment pattern: it provides clean separation between web and agent infrastructure, allows independent certificate management for the AGTP endpoint, and is consistent with service-specific subdomain conventions. An organization with an AGTP subdomain *SHOULD* also configure their primary domain to redirect AGTP requests:

agtp://acme.tld/agents/customer-service
→ 301 → agtp://agtp.acme.tld/agents/customer-service

6.1.8. The /agents/ Reserved Path Prefix

The path prefix /agents/ is reserved in all agtp:// URIs for agent namespace operations. Implementations *MUST* support this prefix. The registry root at /agents (no trailing label) resolves to the Agent Namespace Document (see Section 5.4).

6.1.9. Collision Prevention

agtp://acme.tld/agents/customer-service and agtp://chrishood.tld/agents/customer-service are distinct and non-colliding because the domain component is the trust root. No two agents can share a canonical URI if their org domains differ. Within a single org domain, the governance platform enforces uniqueness of agent labels at registration time. Infrastructure *MUST* use the canonical Agent-ID for all routing, logging, and attribution operations. Human-readable URIs are a display and discovery layer only.

6.1.10. IANA Considerations for the agtp:// URI Scheme

This document proposes registration of the agtp:// URI scheme with IANA per [RFC7595]. Registration template:

URI scheme name: agtp

Status: Permanent

URI scheme syntax: agtp://[canonical-agent-id] or
agtp://[domain.tld]/agents/[label]

URI scheme semantics: Identifies an AI agent or agent namespace operating over the Agent Transfer Protocol. The canonical form uses a 256-bit hex-encoded cryptographic identifier derived from the agent's Birth Certificate. The domain-anchored form uses a verified or asserted organization domain with a reserved /agents/ path prefix.

Applications/protocols that use this URI scheme: Agent Transfer Protocol (this document)

Interoperability considerations: Domain-anchored URIs are the recommended human-readable form. Implementations *MUST* accept canonical identifiers and *SHOULD* support domain-anchored resolution. File extensions *MUST NOT* appear in agtp:// URIs.

Contact: Chris Hood, chris@nomotic.ai

References: This document

The agtp:// URI scheme registration is open and unencumbered. No intellectual property claims apply to the URI scheme itself.

6.2. Trust Tier Summary

Trust Tier	Verification	Package Required	DNS Record Required	Registry Visible
1 - Verified	DNS challenge per [RFC8555]	.nomo	Yes (_agtp TXT)	Yes
2 - Org-Asserted	None	.agent or .nomo	No	Yes (with warning)
3 - Experimental	None	Any	No	No

Table 7: AGTP Trust Tier Summary

6.3. Agent Namespace Document

6.3.1. Purpose and Scope

The Agent Namespace Document is the index of all Active agents registered under an organization's governance zone. It is returned in response to a request targeting the /agents path:

agtp://acme.tld/agents
agtp://agtp.acme.tld/agents

The Agent Namespace Document is not a manually editable file. It is generated and cryptographically signed by the governance platform each time the registry changes. Any Namespace Document that fails signature verification *MUST* be rejected by the requesting party.

6.3.2. Document Schema

```
{
  "document_type": "agtp-namespace",
  "schema_version": "1.0",
  "org_domain": "acme.tld",
  "governance_zone": "zone:acme-internal",
  "generated_at": "2026-03-20T14:00:00Z",
  "signature": {
    "algorithm": "ES256",
    "key_id": "agtp-gov-key-acme-01",
    "value": "[base64-encoded-signature]"
  },
  "agents": [
    {
      "agent_label": "customer-service",
      "canonical_id": "3a9f2c1d8b7e4a6f...",
      "lifecycle_state": "Active",
      "trust_tier": 1,
      "cert_status": "Active",
      "manifest_uri": "agtp://agtp.acme.tld/agents/customer-service",
      "activated_at": "2026-01-15T09:00:00Z",
      "last_updated": "2026-03-01T11:30:00Z"
    }
  ],
  "total_active": 1,
  "namespace_cert_fingerprint": "b2c4d6e8..."
}
```

Figure 3: Agent Namespace Document Schema

The agents array *MUST* include only agents in Active lifecycle state. Suspended, Revoked, and Deprecated agents *MUST NOT* appear in the Namespace Document.

6.3.3. Integrity and Freshness

The Namespace Document *MUST* include a generated_at timestamp. Implementations *SHOULD* treat Namespace Documents older than a configurable freshness threshold (default: 300 seconds) as stale and re-request. The governance platform *MUST* re-sign the Namespace Document within 60 seconds of any registry change.

The signature covers the entire document including generated_at. Replaying an older signed Namespace Document to conceal a revocation event is a known attack vector; implementations *MUST* reject Namespace Documents with a generated_at timestamp older than the freshness threshold.

6.4. Agent Manifest Document and the .agtp Format

6.4.1. Purpose and Scope

The Agent Manifest Document is the protocol’s canonical representation of a specific agent’s identity, status, and behavioral scope. It is returned in response to any AGTP URI resolution request targeting a specific agent:

```
agtp://acme.tld/agents/customer-service
agtp://acme.tld/agents/customer-service?format=json
agtp://acme.tld/agents/customer-service?format=manifest
```

The manifest is derived from the embedded manifest inside the agent’s .agent or .nomo package. It is not a separate file that can be independently modified. The governance platform *MUST* verify the package integrity hash before extracting and serving the manifest.

6.4.2. The Three Document Formats and Their Relationship

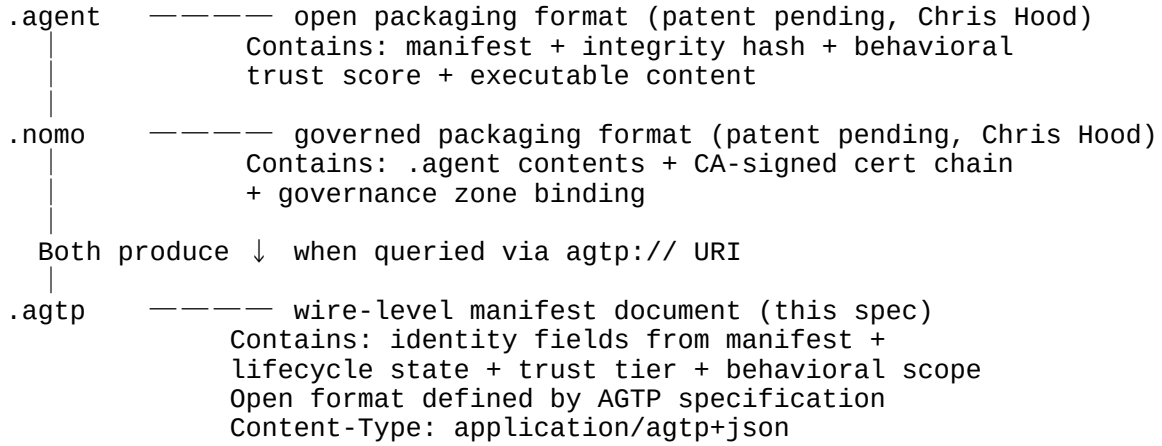

Figure 4: Relationship Between AGTP File Formats

The .agtp format is the protocol's own document type. It is what gets transmitted over the wire. The .agent and .nomo formats are what get deployed. An AGTP implementation is not required to understand .agent or .nomo packaging internals; it is only required to produce and consume .agtp manifest documents.

Additional packaging formats *MAY* be defined by third parties, provided they can produce conformant .agtp manifest documents when queried. Such formats *MUST* implement the integrity verification requirement: the manifest *MUST* be derived from a verified package, not from an independently stored or editable file.

6.4.3. Agent Manifest Document Schema

The following fields are *REQUIRED* in all Agent Manifest Documents:

```
{
  "document_type": "agtp-manifest",
  "schema_version": "1.0",
  "manifest_uri": "agtp://agtp.acme.tld/agents/customer-service",
  "canonical_id": "3a9f2c1d8b7e4a6f0c2d5e9b1a3f7c0d...",
  "agent_label": "customer-service",
  "org_domain": "acme.tld",
  "governance_zone": "zone:acme-internal",
  "trust_tier": 1,
  "package_format": "nomo",
  "package_integrity_hash": "sha256:[hash]",
  "lifecycle_state": "Active",
  "cert_status": "Active",
  "principal_org": "Acme Corporation",
  "activated_at": "2026-01-15T09:00:00Z",
  "behavioral_trust_score": 0.94,
  "authority_scope_categories": [
    "documents:query",
    "knowledge:query",
    "calendar:book",
    "escalation:route"
  ],
  "supported_methods": [
    "QUERY", "SUMMARIZE", "BOOK", "SCHEDULE",
    "ESCALATE", "NOTIFY", "CONFIRM"
  ],
  "job_description": "Handles customer service requests.",
  "signature": {
    "algorithm": "ES256",
    "key_id": "agtp-gov-key-acme-01",
    "value": "[base64-encoded-signature]"
  }
}
```

Figure 5: Agent Manifest Document - Required Fields

The following fields are *RECOMMENDED*:

```
{
  "version": "2.1.0",
  "last_updated": "2026-03-01T11:30:00Z",
  "escalation_policy": "route-to-human-on-scope-limit",
  "delegation_permitted": false,
  "max_delegation_depth": 0,
  "audit_log_uri": "agtp://agtp.acme.tld/audit/customer-service",
  "dns_anchor_record": "_agtp.acme.tld TXT agtp-zone=...",
  "cert_fingerprint": "b2c4d6e8..."
}
```

Figure 6: Agent Manifest Document - Recommended Fields

The following fields are *REQUIRED* when trust_tier is 2:

```
{
  "trust_warning": "org-label-unverified",
  "trust_tier_explanation": "Org label not DNS-verified."
}
```

Figure 7: Agent Manifest Document - Required Fields for Trust Tier 2

6.4.4. What the Manifest Exposes and Does Not Expose

The Agent Manifest Document *MUST* expose:

* The agent’s identity (canonical ID, label, org, governance zone)
* The agent’s current operational status (lifecycle state)
* The agent’s authority scope categories
* The agent’s supported method vocabulary
* The agent’s behavioral trust score
* The agent’s birth certificate fields (activated_at, principal_org)
* The agent’s trust tier and any associated trust warnings

The Agent Manifest Document *MUST NOT* expose:

* Executable code, scripts, or logic
* Model weights or configurations
* Internal API keys or credentials
* Specific authority scope grant tokens
* Session history or prior action logs

No AGTP URI resolution path, including any query parameter combination, *MUST* return package contents beyond the manifest schema defined in this section.

6.4.5. Manifest Tamper-Proofing

The tamper-proof guarantee rests on two mechanisms:

1. *Package integrity hash:* Any modification to the package or its embedded manifest invalidates the hash. The governance platform *MUST* verify this hash before extracting the manifest.

2. *Document signature:* The governance platform signs the extracted manifest before serving it. The signature covers the full document including the package_integrity_hash field, creating a verifiable chain from the served document back to the registered package.

A manifest document that fails either verification step *MUST* be rejected, *MUST NOT* be served, and the failure *MUST* be logged.

6.5. Browser and Human-Facing Interaction Model

6.5.1. The Separation of Discovery and Execution

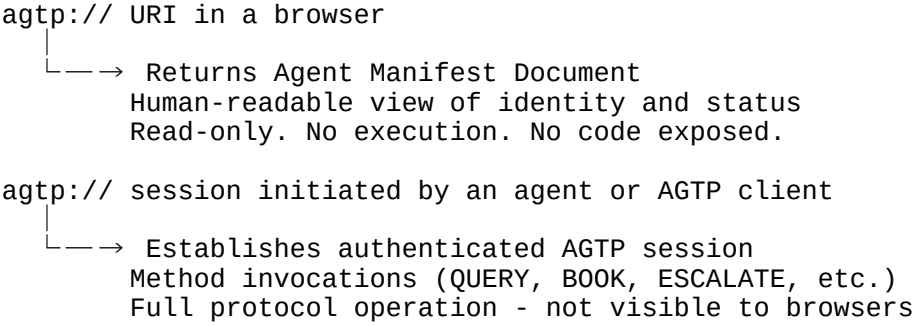

Figure 8: AGTP URI Use by Audience

The analogy to existing protocol conventions is direct. A mailto: URI surfaces an address and hands off to a mail client; SMTP carries the actual messages. Similarly, an agtp:// URI surfaces identity and status; AGTP carries agent traffic. Browsers do not become AGTP clients by following an agtp:// link.

6.5.2. Browser Behavior for agtp:// URIs

Browsers that encounter an agtp:// URI *SHOULD* behave as follows:

1. If a registered AGTP client is present (OS protocol handler), hand off the URI to that client.

2. If the browser supports agtp:// natively or via extension, render the returned Agent Manifest Document as a structured human-readable page. The rendered view *MUST* surface the trust tier indicator prominently, following the visual convention established for TLS trust in the browser chrome.

3. If neither condition applies, the browser *MAY* fall back to a gateway that translates between https:// and agtp://. The gateway *MUST* preserve all signature and trust tier fields.

6.5.3. Human-Readable Manifest View

When an Agent Manifest Document is rendered for human consumption, the following fields *MUST* be prominently displayed:

* Agent label and org domain
* Trust tier indicator (visual distinction between Tier 1, 2, and 3)
* Lifecycle state (Active / Suspended / Revoked / Deprecated)
* Job description
* Principal organization
* Activation date
* Behavioral trust score
* Authority scope categories (in human-readable form)

6.5.4. AGTP Status Sub-Resource

Implementations *SHOULD* support a status sub-path:

agtp://acme.tld/agents/customer-service/status

```
{
  "document_type": "agtp-status",
  "canonical_id": "3a9f2c1d8b7e4a6f...",
  "agent_label": "customer-service",
  "org_domain": "acme.tld",
  "lifecycle_state": "Active",
  "cert_status": "Active",
  "last_action_method": "QUERY",
  "last_action_timestamp": "2026-03-20T13:58:22Z",
  "active_session_count": 3,
  "pending_escalations": 0,
  "generated_at": "2026-03-20T14:00:00Z"
}
```

Figure 9: AGTP Status Sub-Resource Response

The active_session_count field *SHOULD* only be included if the requester has appropriate observability permissions for the governance zone.

6.6. Web3 Interaction Considerations

6.6.1. The .agent TLD Collision

The string .agent is claimed as a Web3 top-level domain by at least one blockchain-based naming system. This creates an ambiguity risk: a URI of the form agtp://customer-service.agent could be interpreted as either an AGTP agent namespace or a Web3 name resolution request. This specification explicitly prohibits .agent and .nomo from appearing in the hostname position of agtp:// URIs (see Section 5.1).

6.6.2. Web3 Trust Anchors

AGTP’s default trust model is PKI-based: X.509 certificates, DNS ownership verification, and CA-anchored identity. Web3 naming systems provide an alternative identity model based on blockchain address ownership. A full Web3 interoperability specification is out of scope for this document. The following framework *MAY* be used by implementations wishing to bridge Web3 and AGTP identity.

The resolution_layer field in the Agent Manifest Document declares the identity anchoring mechanism used for the agent’s registration:

Value	Meaning
dns	Standard DNS ownership verification (default)
pki	PKI certificate chain without DNS challenge
web3-ens	Ethereum Name Service
web3-unstoppable	Unstoppable Domains
agtp-registry	Direct registration with AGTP governance platform

Table 8: resolution_layer Field Values

Implementations that encounter a resolution_layer value of web3-ens or web3-unstoppable *MUST* treat the agent as Trust Tier 2 (Org-Asserted) unless a formal AGTP-Web3 Bridge specification has been published and implemented. Full Web3 interoperability is addressed in a future companion document [AGTP-WEB3].

6.7. Agent Registration Process

6.7.1. Overview

An agent cannot participate in AGTP until it has been issued an Agent Birth Certificate by a governance platform and assigned a canonical Agent-ID derived from that certificate. Canonical Agent-IDs are issued through the ACTIVATE transaction; they are never self-declared.

The Birth Certificate is the genesis record of an agent’s legal existence within the AGTP ecosystem. Its relationship to the canonical Agent-ID is analogous to the relationship between a government-issued birth registration and a social security number: the birth event produces a permanent, authoritative identity record, and a durable identifier is derived from it. The identifier follows the agent for its entire lifecycle, including after revocation. It is never reissued to another agent.

Any AGTP infrastructure component *MUST* reject requests carrying an Agent-ID that does not resolve to a Birth Certificate record in an Active lifecycle state in a reachable registry.

6.7.2. Birth Certificate Contents

The Agent Birth Certificate is issued by the governance platform at ACTIVATE time and contains the following fields:

Field	Required	Description
agent_id	*MUST*	Unique identifier for the agent
owner	*MUST*	Human or team responsible for this agent
archetype	*MUST*	Behavioral category (see archetypes below)
governance_zone	*MUST*	Environment context (development, staging, production)
scope	*MUST*	Authorized action types
issued_at	*MUST*	Timestamp of issuance
certificate_hash	*MUST*	Cryptographic fingerprint - basis for canonical Agent-ID
signature	*MUST*	Signed with the org’s governance key
package_ref	*SHOULD*	Reference to the .agent or .nomo package
trust_tier	*MUST*	Registration tier (1, 2, or 3)
org_domain	*MUST*	The verified or asserted org domain

Table 9: Agent Birth Certificate Fields

6.7.3. Agent Archetypes

The archetype field classifies the agent’s behavioral category. Archetypes inform scope enforcement and observability tooling; an executor archetype agent exhibiting read-only query patterns, or a monitor archetype agent attempting booking operations, are anomaly signals. The archetype field does not restrict scope enforcement, Authority-Scope headers govern actual permissions at the protocol

level. Archetypes are a classification and observability signal, not a security boundary.

Archetype	Description	Typical Scope
assistant	Conversational agent, read-heavy	documents:query, knowledge:query
analyst	Data analysis, read and aggregate	data:read, data:aggregate
executor	Takes real-world actions, write-heavy	booking:*, payments:confirm
orchestrator	Manages other agents	delegation:*, agents:*
monitor	Observational only	telemetry:read, logs:read

Table 10: Agent Archetypes

6.7.4. Birth Certificate to AGTP Header Mapping

Birth Certificate fields map directly to AGTP protocol headers on every request. This mapping is the mechanism by which static identity (the Birth Certificate) becomes runtime identity (the protocol session):

Birth Certificate Field	AGTP Protocol Header
agent_id	Agent-ID
owner	Principal-ID
scope	Authority-Scope
certificate_hash	Basis for canonical Agent-ID

Table 11: Birth Certificate to AGTP Header Mapping

The canonical Agent-ID is derived from the certificate_hash. This chain, package integrity hash → certificate hash → canonical Agent-ID, ensures that the identifier carried in the Agent-ID header on

every AGTP request is traceable back to the original Birth Certificate and the human principal who authorized the agent's creation.

6.7.5. Registration Tiers

Tier 1 Registration (Verified, DNS-anchored):

Required for agents carrying Authority-Scope beyond read-only query operations, or participating in delegation chains, financial transactions, or multi-agent collaboration with external organizations.

Requirements:

- Registrant demonstrates DNS control over the claimed org_domain via DNS challenge per [RFC8555]
- Agent package must be in .nomo governed format
- Package must include a valid CA-signed certificate chain
- Governance platform issues Birth Certificate and canonical Agent-ID after verifying DNS challenge and validating the package cert chain

Tier 2 Registration (Org-Asserted):

For agents operating within a single organization's internal infrastructure, or where DNS verification is not yet completed.

Requirements:

- Org label is declared but DNS ownership is not verified
- Agent package may be .agent or .nomo format
- Governance platform issues Birth Certificate after validating package integrity hash
- Birth Certificate and Manifest *MUST* include trust_tier: 2 and trust_warning: "org-label-unverified"
- Authority scope *MUST* be restricted at the SEP layer until upgraded to Tier 1

Tier 3 Registration (Experimental):

For development and testing environments only.

Requirements:

* Agent label *MUST* carry X- prefix
* Not published to the public AGTP registry
* *MUST NOT* be deployed in production environments
* Governance platform issues a locally-scoped Birth Certificate

6.7.6. Registration Lifecycle

1. PACKAGE
   Author creates .agent or .nomo package containing:
   - Embedded manifest (agent_label, job_description, authority_scope_categories, supported_methods, behavioral_trust_score)
   - Integrity hash of all package contents
   - For .nomo: CA-signed certificate chain

2. SUBMIT (ACTIVATE transaction)
   Registrant submits ACTIVATE request to governance endpoint:
   - Package file (.agent or .nomo)
   - Proposed agent label and org domain
   - Owner identity (maps to Birth Certificate owner field)
   - Archetype declaration
   - For Tier 1: DNS challenge token

3. VALIDATE (governance platform)
   Governance platform:
   - Verifies package integrity hash
   - For .nomo: validates certificate chain
   - For Tier 1: verifies DNS challenge against _agtp.[org_domain] TXT record
   - Checks proposed label for uniqueness within org namespace

4. ISSUE (Birth Certificate and canonical Agent-ID assigned)
   Governance platform:
   - Issues Agent Birth Certificate with all fields populated
   - Derives canonical Agent-ID from certificate_hash
   - Creates registry record with Active lifecycle state
   - Records genesis audit entry in immutable audit log (genesis record includes full Birth Certificate)
   - Publishes agent to Namespace Document (triggers Namespace Document re-signing)

   The Birth Certificate is delivered to the registrant.
   It is the permanent record of the agent’s genesis.
   Loss of the Birth Certificate does not invalidate the agent;

the certificate_hash remains the authoritative identity anchor.

5. ACTIVE
   Agent enters Active lifecycle state.
   Canonical Agent-ID is valid for AGTP protocol sessions.
   agtp://[org_domain]/agents/[label] resolves to manifest
   derived from the Birth Certificate.

6. LIFECYCLE EVENTS (post-activation)
   SUSPEND: Agent temporarily inactive. Manifest returns 503.
   Birth Certificate and canonical ID remain valid.
   Initiated by trust violation or human decision.
   REINSTATE: Human-authorized return to Active state.
   Birth Certificate unchanged. Reinstatement
   recorded in audit trail.
   REVOKE: Agent permanently deactivated. Manifest returns 410.
   Birth Certificate archived. Canonical ID retired
   permanently and never reissued.
   DEPRECATE: Controlled end-of-life. Manifest returns 410 with
   successor_agent field if applicable. Birth
   Certificate retained per Section 8.5 retention policy.

Figure 10: AGTP Agent Registration Lifecycle

6.7.7. Governance Tokens and Runtime Authorization

Following successful registration, the agent’s Birth Certificate is the static identity anchor. Runtime authorization for specific actions is carried by Governance Tokens: signed, time-limited JWT artifacts issued by the governance platform encoding a specific governance verdict (ALLOW, DENY) for a specific action.

Governance Tokens *MUST NOT* be reused. Each action requires a fresh evaluation and a fresh token. Default TTL is 30 seconds. The token’s agent_id field *MUST* match the canonical Agent-ID from the Birth Certificate. Tokens that fail this validation *MUST* be rejected and the failure *MUST* be logged.

The relationship between Birth Certificate and Governance Token parallels the relationship between a passport and a visa: the passport establishes persistent identity; the visa encodes a specific time-bounded permission. Holding a passport does not imply holding any particular visa.

6.7.8. Friendly Name Availability and Re-Registration

An agent label becomes available for re-registration 90 days after its associated agent enters Revoked or Deprecated lifecycle state. The canonical Agent-ID and Birth Certificate are permanently archived. The canonical Agent-ID *MUST NOT* be reissued under any circumstances, including re-registration of the same label by the same organization. This policy prevents ID reuse attacks in which a newly registered agent inherits the trust history of a revoked predecessor.

7. Method Definitions

7.1. Design Philosophy

AGTP methods are intent verbs, not resource operations. Each method expresses what an agent is trying to accomplish. Method names are uppercase ASCII strings. Methods that modify state are NOT idempotent by default unless explicitly marked. All methods accept a context parameter carrying agent session state. Requirement language follows [RFC2119].

7.2. Core Methods

7.2.1. QUERY

Purpose: Semantic data retrieval. The agent specifies what it needs to know, not where to find it. Distinguished from HTTP GET by expressing an information need rather than retrieving a known resource at a known location.

Parameter	Required	Description
intent	*MUST*	Natural language or structured expression of the information need
scope	*SHOULD*	Data domains or sources to include or exclude
format	*MAY*	Desired response format: structured, natural, raw
confidence_threshold	*MAY*	Minimum confidence score for included results (0.0-1.0)
context	*MAY*	Session context for disambiguation

Table 12: QUERY Parameters

Response: Result set with confidence scores per item. Server *SHOULD* indicate provenance of each result. Idempotent: Yes.

7.2.2. SUMMARIZE

Purpose: Request a concise synthesis of provided content or a referenced resource. The agent is requesting a cognitive operation on data, not retrieving data.

Parameter	Required	Description
source	*MUST*	Content inline (up to implementation limit) or URI reference
length	*SHOULD*	Target summary length: brief, standard, detailed
focus	*MAY*	Aspect to emphasize in the summary
format	*MAY*	Output format: bullets, prose, structured
audience	*MAY*	Intended reader context, for calibrating complexity

Table 13: SUMMARIZE Parameters

Response: Summary content with a source_hash and a confidence score. Idempotent: Yes.

7.2.3. BOOK

Purpose: Reserve a resource, time slot, seat, or allocation on behalf of the agent’s principal. State-modifying. Notable error codes: 409 Conflict (resource unavailable), 451 Scope Violation (principal not authorized for this resource type).

Parameter	Required	Description
resource_id	*MUST*	Identifier of the resource to reserve
principal_id	*MUST*	The human or system on whose behalf the booking is made
time_slot	*MUST* (if time-based)	ISO 8601 datetime or range
quantity	*MAY*	Number of units to reserve
options	*MAY*	Resource-specific booking parameters
confirm_immediately	*MAY*	Boolean; if false, creates a hold pending confirmation

Table 14: BOOK Parameters

Response: Booking confirmation with booking_id, status (confirmed / held), and expiry timestamp if a hold. Idempotent: No.

7.2.4. SCHEDULE

Purpose: Define a sequence of actions, method calls, or events to be executed at specified times or in response to specified triggers. Creates a durable plan, not an immediate execution.

Parameter	Required	Description
steps	*MUST*	Ordered list of AGTP method calls with parameters
trigger	*MUST*	immediate, datetime, event, or condition
trigger_value	*MUST* (if not immediate)	Datetime, event name, or condition expression
on_failure	*SHOULD*	Behavior on step failure: abort, skip, retry, escalate
notify	*MAY*	Notification targets on completion or failure

Table 15: SCHEDULE Parameters

Response: Schedule record with schedule_id, confirmed steps, and next execution timestamp. Idempotent: No.

7.2.5. LEARN

Purpose: Update the agent's session context, knowledge state, or persistent memory. An explicit context write where the agent asserts that something should be retained.

Parameter	Required	Description
content	*MUST*	Information to be learned (structured or unstructured)
scope	*MUST*	session (ephemeral), principal (persists for principal), global (shared)
category	*SHOULD*	Semantic category for retrieval optimization
confidence	*MAY*	Agent’s confidence in the content (0.0-1.0)
source	*MAY*	Provenance of the learned content
ttl	*MAY*	Expiry for the learned content

Table 16: LEARN Parameters

Response: Confirmation with learn_id and effective scope.
Idempotent: No.

7.2.6. DELEGATE

Purpose: Transfer execution of a task or method to a sub-agent or downstream system. Initiates a new AGTP session on behalf of the delegating agent, carrying forward authority lineage.

Parameter	Required	Description
target_agent_id	*MUST*	Identifier of the agent to delegate to
task	*MUST*	AGTP method call (or sequence) to execute
authority_scope	*MUST*	Scope granted to sub-agent *MUST* be a strict subset of delegating agent's scope
delegation_token	*MUST*	Signed token proving delegation authority
callback	*SHOULD*	AGTP endpoint for result delivery
deadline	*MAY*	Maximum time for task completion

Table 17: DELEGATE Parameters

Security note: the authority_scope in a DELEGATE request *MUST NOT* exceed the delegating agent's own Authority-Scope. Servers *MUST* enforce this and *MUST* return 451 Scope Violation if violated. This is the protocol-level defense against authority laundering. Idempotent: No.

7.2.7. COLLABORATE

Purpose: Initiate a multi-agent coordinated task where two or more agents work in parallel or in defined roles toward a shared goal. Unlike DELEGATE (hierarchical), COLLABORATE is peer-to-peer.

Parameter	Required	Description
collaborators	*MUST*	List of Agent-IDs invited to collaborate
objective	*MUST*	Shared goal expressed as a task description or structured specification
role_assignments	*SHOULD*	Map of Agent-IDs to roles within the collaboration
coordination_model	*SHOULD*	parallel, sequential, or consensus
result_aggregation	*MAY*	How results from collaborators are combined

Table 18: COLLABORATE Parameters

Response: Collaboration session receipt with collaboration_id. Each collaborator receives an AGTP NOTIFY to join. Idempotent: No.

7.2.8. CONFIRM

Purpose: Explicit acknowledgment of a prior action, state, or data item. Creates a signed attestation record.

Parameter	Required	Description
target_id	*MUST*	ID of the action, booking, schedule, or item being confirmed
status	*MUST*	accepted, rejected, or deferred
reason	*SHOULD* (if rejected/ deferred)	Explanation of the decision
attestation	*MAY*	Agent-signed confirmation payload for audit

Table 19: CONFIRM Parameters

Response: Confirmation receipt with timestamp and attestation_id.
Idempotent: Yes.

7.2.9. ESCALATE

Purpose: Route a task, decision, or exception to a human principal or higher-authority agent when the current agent cannot or should not proceed. ESCALATE is the protocol-level expression of meaningful friction in AI systems as a first-class method.

Parameter	Required	Description
task_id	*MUST*	The task or method invocation triggering escalation
reason	*MUST*	Structured reason: confidence_threshold, scope_limit, ethical_flag, ambiguous_instruction, resource_unavailable
context	*MUST*	Full context needed for the escalation recipient to act
priority	*SHOULD*	urgent, normal, or low
recipient	*MAY*	Specific human or agent to escalate to; if absent, routes to default handler
deadline	*MAY*	Time by which a response is needed

Table 20: ESCALATE Parameters

Response: Escalation receipt with escalation_id and routing confirmation. The escalated task is paused until resolved via CONFIRM. Idempotent: Yes. An agent that escalates appropriately is functioning correctly. Governance frameworks built on AGTP can use escalation frequency and reason codes as observability signals for systemic issues.

7.2.10. NOTIFY

Purpose: Asynchronous push of information from an agent to a recipient. Does not expect a response. Fire-and-forget. Delivery confirmation (if required) returned via a subsequent CONFIRM from the recipient.

Parameter	Required	Description
recipient	*MUST*	Target Agent-ID, human endpoint, or broadcast group
content	*MUST*	Notification payload
urgency	*SHOULD*	critical, informational, or background
delivery_guarantee	*MAY*	at_most_once, at_least_once, or exactly_once
expiry	*MAY*	Timestamp after which the notification should not be delivered

Table 21: NOTIFY Parameters

Response: Delivery receipt with notification_id. Idempotent: No.

7.2.11. DESCRIBE

Purpose: Return the operational capabilities of a known agent endpoint. The requesting agent specifies what capability dimensions it needs to evaluate; the server returns a structured Capability Document. Used for pre-task negotiation before committing to DELEGATE or COLLABORATE. If capability_domains is omitted, the server *SHOULD* return all supported domains. Category: ACQUIRE.

Parameter	Required	Description
capability_domains	*SHOULD*	Comma-separated domains to return: methods, modalities, tools, version, budget, zones. If omitted, server *SHOULD* return all.
version_min	*MAY*	Minimum acceptable version for capability negotiation.
context	*MAY*	Session context for capability filtering.

Table 22: DESCRIBE Parameters

Response: Capability Document with the following structure:

```
{
  "supported_methods": ["QUERY", "SUMMARIZE", "DESCRIBE"],
  "modalities": ["text", "image", "streaming"],
  "tools": ["web_search", "code_execute"],
  "version": "2.0.0",
  "version_min_satisfied": true,
  "behavioral_trust_score": 0.94,
  "budget_units_accepted": ["tokens", "compute-seconds"],
  "zones_accepted": ["zone:internal", "zone:partner"]
}
```

Idempotent: Yes. Primary error codes: 404, 422.

7.2.12. SUSPEND

Purpose: Pause a specific active session workflow in a recoverable state. Issues a resumption nonce the requesting agent uses to resume the session. Method-level SUSPEND is session-scoped and does not affect registry lifecycle state or Birth Certificate validity. The distinction between method-level SUSPEND and lifecycle SUSPEND (Section 6.7.6) is architectural: method-level SUSPEND is a workflow primitive; lifecycle SUSPEND is an administrative action on the agent’s registry record. Category: ORCHESTRATE.

Parameter	Required	Description
session_id	*MUST*	The session to suspend.
reason	*SHOULD*	Structured reason: awaiting_input, resource_limit, scheduled_pause, external_dependency.
resume_by	*MAY*	ISO 8601 deadline for resumption. If exceeded without RESUME, session transitions to expired.
checkpoint	*MAY*	Agent-provided state snapshot for resumption context. Stored by server for duration of suspension.

Table 23: SUSPEND Parameters

Response: Suspension receipt with the following structure:

```
{
  "suspension_id": "susp-0042",
  "session_id": "sess-a1b2c3d4",
  "resumption_nonce": "[128-bit random value, base64url]",
  "resume_by": "2026-04-15T09:00:00Z",
  "status": "suspended"
}
```

The resumption_nonce *MUST* be a cryptographically random 128-bit value encoded as base64url. It is single-use: once presented to resume a session, the nonce is invalidated and *MUST NOT* be accepted again. Idempotent: No. Primary error codes: 404, 408.

Servers MUST generate nonces with at least 128 bits of entropy using a CSPRNG.

7.3. Method Summary Table

Method	Intent	State-Modifying	Idempotent	Primary Error Codes
QUERY	Retrieve information	No	Yes	404, 422

SUMMARIZE	Synthesize content	No	Yes	400, 422
BOOK	Reserve a resource	Yes	No	409, 451
SCHEDULE	Plan future actions	Yes	No	400, 409
LEARN	Update agent context	Yes	No	400, 403
DELEGATE	Transfer task to sub-agent	Yes	No	403, 451, 551
COLLABORATE	Coordinate peer agents	Yes	No	404, 403
CONFIRM	Attest to a prior action	Yes	Yes	404, 400
ESCALATE	Defer to human/ authority	Yes	Yes	404
NOTIFY	Push information	No	No	400, 404
DESCRIBE	Retrieve endpoint capabilities	No	Yes	404, 422
SUSPEND	Pause session workflow	Yes	No	404, 408

Table 24: AGTP Core Method Summary

7.4. Method Registry and Extensibility

AGTP defines a formal Method Registry maintained by IANA (see Section 8.2). Any party may submit a new method for registration. The registration procedure is Expert Review, and registration *MUST* be accompanied by a published specification, at minimum an IETF Internet-Draft or equivalent publicly available document. Registered methods *MUST*:

1. Have a unique uppercase ASCII name

2. Define required and optional parameters

3. Define expected response structure

4. Specify idempotency behavior

5. Specify applicable error codes

6. Include a security considerations section

7. Be accompanied by a published reference specification (Internet-Draft or RFC)

Experimental methods *MAY* be used prior to registration using the X- prefix convention (e.g., X-NEGOTIATE). Experimental methods *MUST NOT* be used in production deployments without registration.

Capability negotiation occurs during session establishment. The server returns a Supported-Methods header listing the methods it implements. Clients *SHOULD* check this list before invoking non-core methods.

QUOTE is defined as a Tier 2 Standard Extended Method in [AGTP-METHODS]. QUOTE provides pre-flight cost estimation for a proposed method invocation: the requesting agent submits a proposed method call; the server returns a Cost-Estimate response without executing the method. Servers supporting budget negotiation via the Budget-Limit header *SHOULD* implement QUOTE to enable agents to validate cost before committing to execution. Servers that implement QUOTE *MUST* list it in the Supported-Methods response header at session establishment.

7.5. Extended Method Vocabulary and Industry Profiles

7.5.1. Three-Tier Method Architecture

The AGTP method vocabulary is organized into three tiers reflecting different levels of universality, specificity, and domain relevance.

Tier 1. Core Methods (defined in Section 6.2): The baseline vocabulary required for AGTP compliance. Every conformant AGTP implementation *MUST* support all Tier 1 methods.

Tier 2. Standard Extended Methods: Registered in the IANA AGTP

Method Registry and available for use in any AGTP implementation. Not required for baseline compliance but *SHOULD* be implemented where their semantics apply. Defined in [AGTP-METHODS].

Tier 3. Industry Profile Methods: Domain-specific method sets defined and registered by industry communities as named AGTP profiles. Valid within deployments that declare support for the relevant profile. Not required in general-purpose implementations.

7.5.2. Method Category Taxonomy

All AGTP methods are organized into five categories:

ACQUIRE: Retrieve data, resources, or state without modifying it. Typically idempotent; no state modification.

COMPUTE: Process, transform, or analyze information and produce a derived result. Typically idempotent given the same input.

TRANSACT: Perform state-changing operations with external systems, resources, or records. Not idempotent by default; subject to reversibility classification.

COMMUNICATE: Send information, notifications, or signals to recipients. Fire-and- forget or confirm-receipt delivery models.

ORCHESTRATE: Coordinate, sequence, or manage multiple agents, tasks, or workflows. May spawn sub-agents or sessions; delegation chain semantics apply.

Core Method	Category
QUERY	Acquire
SUMMARIZE	Compute
BOOK	Transact
SCHEDULE	Orchestrate
LEARN	Compute
DELEGATE	Orchestrate
COLLABORATE	Orchestrate
CONFIRM	Transact
ESCALATE	Orchestrate
NOTIFY	Communicate
DESCRIBE	Acquire
SUSPEND	Orchestrate

Table 25: Core Method Category Mapping

7.5.3. Standard Extended Methods (Tier 2)

The following methods constitute the initial Tier 2 registration set, defined in [AGTP-METHODS]. Listed here by category with brief semantic definitions; full parameter specifications are in the companion document.

ACQUIRE category: FETCH, SEARCH, SCAN, PULL, IMPORT, FIND.

COMPUTE category: EXTRACT, FILTER, VALIDATE, TRANSFORM, TRANSLATE, NORMALIZE, PREDICT, RANK, MAP.

TRANSACT category: REGISTER, SUBMIT, TRANSFER, PURCHASE, SIGN, MERGE, LINK, LOG, SYNC, PUBLISH.

COMMUNICATE category: REPLY, SEND, REPORT.

ORCHESTRATE category: MONITOR, ROUTE, RETRY, PAUSE, RESUME, RUN, CHECK.

Notable constraints: PURCHASE *MUST* carry explicit principal_id and scope enforcement; 451 Scope Violation applies if payments:purchase is not in the agent's Authority-Scope. RUN requires explicit procedure_id parameter; implementations *MUST NOT* accept free-form execution strings.

7.5.4. Short-Form and Industry-Inspired Methods

A set of short-form verb methods, e.g., SET, TAKE, OPEN, START, CALL, MAKE, TURN, BREAK, are provisionally catalogued as candidates for Tier 2 registration. These verbs are highly context-dependent and their semantics vary significantly across deployment domains.

Short-form methods will be registered individually only when a published companion specification provides unambiguous semantic definitions demonstrably distinct from existing registered methods. Provisional registrations using the X- prefix (e.g., X-SET, X-CALL) are encouraged during the experimentation period.

7.5.5. Industry Profile Method Sets

AGTP recognizes that specific industries require method vocabularies reflecting domain-specific operations that would be inappropriate in a general-purpose standard. Industry profile method sets are defined and registered as named AGTP profiles. A profile is a published companion specification that:

1. Declares a profile name (e.g., agtp-profile-healthcare, agtp-profile-financial, agtp-profile-legaltech)

2. Defines one or more industry-specific methods with full parameter specifications, error codes, and security considerations

3. Specifies which Tier 1 and Tier 2 methods are REQUIRED, RECOMMENDED, or NOT APPLICABLE within the profile

4. Addresses regulatory or compliance considerations specific to the domain (e.g., HIPAA for healthcare, PCI-DSS for financial services)

Illustrative examples of potential industry profile methods (not yet registered; listed for directional purposes only):

Healthcare: PRESCRIBE, AUTHORIZE, REFER, DISPENSE, TRIAGE, CONSENT, REDACT

Financial services: SETTLE, RECONCILE, HEDGE, CLEAR, UNDERWRITE, KYC, AML

Legal and compliance: ATTEST, NOTARIZE, DISCLOSE, REDLINE, EXECUTE, PRESERVE

Infrastructure: PROVISION, DEPROVISION, ROLLBACK, SNAPSHOT, FAILOVER

Industry communities are encouraged to develop and submit profile specifications through the IETF process. The IANA AGTP Method Registry will maintain a profile index alongside the core and standard method registries.

7.5.6. Registration Path for New Methods

For Tier 2 Standard Methods: Submit an Internet-Draft to the IETF providing full method specification per Section 6.4. The Designated Expert reviews for semantic uniqueness, clarity, and security considerations.

For Industry Profile Methods: Submit a profile specification to the IETF (or a recognized domain standards body with an established AGTP registry liaison) covering all methods in the profile and profile compliance requirements.

For Experimental Methods: Use the X- prefix without registration. Implementations *MUST NOT* deploy experimental methods in production without completing the registration process. Experimental method names do not reserve the unprefixed name.

The AGTP Method Registry is published at: https://www.iana.org/assignments/agtp-methods/

8. Security Considerations

This section satisfies the mandatory IETF Security Considerations requirement. All AGTP implementations *MUST* address the considerations described here.

8.1. Mandatory TLS

All AGTP connections *MUST* use TLS 1.3 or higher. Implementations *MUST* reject connections using TLS 1.2 or below. Certificate validation follows standard PKI practices per [RFC5280]. Servers *MUST* present a valid certificate.

8.2. Agent Identity Headers and Agent Certificate Extension

Every AGTP request *MUST* include Agent-ID and Principal-ID header fields. In the base specification, these fields are not cryptographically authenticated. They are self-asserted but logged mandatorily for auditability. Implementations *SHOULD* use logging and anomaly detection to identify inconsistencies.

Full cryptographic verification of agent identity and Authority-Scope is provided by the AGTP Agent Certificate extension [AGTP-CERT]. That extension binds Agent-ID, Principal-ID, and Authority-Scope to an X.509 v3 certificate presented during TLS mutual authentication, enabling infrastructure-layer identity and scope verification without application-layer access. Implementers planning deployments that require verified agent identity *SHOULD* plan for the Agent Certificate extension.

> Note: The Agent Certificate extension and the Agent Birth Certificate mechanism may be subject to pending intellectual property claims. See Section 7.7 and the IPR Notice preceding the Abstract for details. The licensor is prepared to grant a royalty-free license to implementers.

Every AGTP server *MUST* log Agent-ID and Principal-ID fields for every request, creating an attributable audit trail even in deployments without the Certificate extension.

8.3. Authority Scope Enforcement

The Authority-Scope header declares what actions the agent is authorized to take. Compliant AGTP servers *MUST* parse the Authority-Scope on every request, return 451 Scope Violation for any method that exceeds declared scope, and log all scope violations for audit purposes. Scope declarations are self-asserted in the base spec, analogous to scope assertions in OAuth 2.0 [RFC6749]. Cryptographically signed and infrastructure-enforced scopes are defined in [AGTP-CERT].

8.4. Threat Model

8.4.1. Agent Spoofing

Threat: A malicious actor forges Agent-ID and Principal-ID headers to impersonate a trusted agent. Base spec mitigation: mandatory logging and anomaly detection. Full mitigation requires [AGTP-CERT].

8.4.2. Authority Laundering

Threat: An agent claims an Authority-Scope broader than what it was granted. Mitigation: server-side scope enforcement; 451 Scope Violation returned and logged. In DELEGATE chains, each hop's scope *MUST* be a strict subset of the delegating agent's scope.

8.4.3. Delegation Chain Poisoning

Threat: A malicious agent inserts itself into a DELEGATE chain. Mitigation: Delegation-Chain headers are logged at each hop. 551 Authority Chain Broken is returned if any chain entry is unverifiable. Full mitigation requires [AGTP-CERT] for signed delegation tokens.

8.4.4. Denial of Service via High-Frequency Agent Traffic

Threat: Agents that are compromised, misconfigured, or adversarial generate extremely high request volumes. Mitigation: 429 Rate Limited status code. Rate limiting *SHOULD* be applied per Agent-ID and per Principal-ID. When [AGTP-CERT] is deployed, per-Agent-ID quotas can be cryptographically tied to verified identity, preventing quota evasion through Agent-ID spoofing.

8.4.5. Session Hijacking

Threat: An attacker intercepts or forges a Session-ID. Mitigation: mandatory TLS protects sessions in transit. Session-IDs *MUST* be cryptographically random with minimum 128 bits of entropy. Servers *MUST* validate that Session-ID, Agent-ID, and TLS client identity are consistent.

8.4.6. Escalation Suppression

Threat: A compromised agent or intermediary suppresses ESCALATE requests, preventing human oversight. Mitigation: compliant implementations *MUST* route ESCALATE requests directly to the declared escalation handler without modification. Intermediaries *MUST NOT* drop, delay, or modify ESCALATE requests. Escalation handlers *SHOULD* implement independent receipt confirmation.

8.4.7. Birth Certificate Spoofing

Threat: A malicious actor fabricates a Birth Certificate to claim a legitimate agent's identity or construct a false identity with elevated trust. Mitigation: Birth Certificates are issued only by governance platforms with verified ownership of the org_domain. In the base spec, mandatory logging provides auditability. Full mitigation requires [AGTP-CERT] for cryptographically bound Birth Certificate verification at the transport layer. Governance platforms *MUST* treat any ACTIVATE request that presents a certificate hash matching an existing registry record as a collision attack and *MUST* reject it.

8.4.8. Domain Transfer Identity Hijacking

Threat: An attacker acquires an expired domain to inherit the agent registry and trust history of prior registrants. Mitigation: agents under an expired domain are automatically Suspended within 24 hours of domain expiry detection. A new owner of the domain *MUST NOT* inherit prior agent registrations. See Section 9.6 for the full domain expiry policy.

8.4.9. Attribution Forgery

Threat: A malicious agent submits a fabricated or replayed Attribution-Record to claim credit for an action it did not perform, or to conceal the true execution context of an action it did perform.

Mitigation: Attribution-Records *MUST* be signed with the agent's governance key. The signature *MUST* cover the full record including the Task-ID, Agent-ID, method, timestamp, and result hash. When [AGTP-CERT] is deployed, the signature is verified at the transport layer against the agent's X.509 certificate. For high-stakes domains, RATS attestation evidence in the Attribution-Record per [RFC9334] provides hardware-rooted proof of execution context that cannot be forged without compromising the attesting environment itself. Attribution-Record signatures *MUST* be verified before the record is admitted to an audit trail. Unverified records *MUST* be logged with a signature_unverified flag and *MUST NOT* be treated as authoritative for compliance purposes.

8.5. Privacy Considerations

Agent identity headers carry information about agent behavior that may be sensitive:

* Agent-ID and Principal-ID together may reveal organizational structure

* Session-ID and Task-ID reveal workflow patterns

* Delegation-Chain reveals multi-agent architecture

AGTP logs containing these fields *MUST* be treated as sensitive operational data. Operators *MUST* implement appropriate access controls, retention limits, and data minimization practices consistent with applicable privacy regulations.

Where privacy-preserving attribution is required, implementations *MAY* use pseudonymous Agent-IDs with a separate trusted resolution service. The architecture for pseudonymous agent identity resolution is reserved for a future companion document.

8.6. Denial-of-Service Considerations

AGTP's agent identity headers provide a mechanism for more precise denial-of-service mitigation than is possible with HTTP. Rate limiting *SHOULD* be applied per Agent-ID and per Principal-ID in addition to per-IP-address controls.

When [AGTP-CERT] is deployed, per-Agent-ID rate limiting can be cryptographically tied to verified agent identity, preventing quota evasion through Agent-ID rotation. Implementations planning high-volume governed agent deployments *SHOULD* plan for [AGTP-CERT] as part of their denial-of-service mitigation strategy.

Additional recommended mitigations: Priority header enforcement (Priority: background requests *SHOULD* have lower rate limit headroom than Priority: critical); per-governance-zone aggregate limits in multi-tenant deployments; and circuit breaker patterns for ESCALATE request floods.

8.7. Intellectual Property Considerations

The core AGTP specification, including all base methods, header fields, status codes, connection model, and IANA registrations defined in this document, is intended for open implementation without royalty obligation.

Certain elements referenced in this document may be subject to pending patent applications by the author, specifically:

* The Agent Certificate extension [AGTP-CERT], which provides cryptographic binding of agent identity and authority scope to AGTP header fields.

* The ACTIVATE method, which provides AGTP-native transmission and activation of governed agent packages.

* The Agent Birth Certificate mechanism (Section 5.7), which provides the genesis identity record and canonical Agent-ID derivation process for AGTP-registered agents.

* The .agent file format specification, an open packaging format for AI agents.

* The .nomo file format specification, a governed packaging format for AI agents with cryptographic governance binding.

Implementers of the core AGTP specification are not affected by any intellectual property claims on these extensions and associated formats.

The licensor is prepared to grant a royalty-free license to implementers for any patent claims that cover contributions in this document and its referenced extensions, consistent with the IETF's IPR framework under [RFC8179].

IPR disclosures have been filed with the IETF Secretariat and are available at: https://datatracker.ietf.org/ipr/

9. IANA Considerations

This document requests the following IANA actions upon advancement to RFC status.

9.1. Port Assignment

Registration of the following service names in the IANA Service Name and Transport Protocol Port Number Registry:

Service Name	Port	Transport	Description
agtp	TBD	TCP	Agent Transfer Protocol over TCP/TLS
agtp-quic	TBD	UDP	Agent Transfer Protocol over QUIC

Table 26: Proposed Port Assignments

9.2. AGTP Method Registry

Establishment of a new IANA registry: Agent Transfer Protocol Methods.

Registry name: Agent Transfer Protocol Methods

Registration procedure: Expert Review per [RFC8126], with the additional requirement that each registration be accompanied by a published specification, at minimum a publicly available Internet-Draft or equivalent document. The Designated Expert *SHOULD* verify that the proposed method name is unique, the reference specification is publicly accessible, and the method definition includes the required fields (parameters, response structure, idempotency, error codes, security considerations).

Reference: This document

Initial registrations:

Method	Status	Reference
QUERY	Permanent	This document, Section 7.2
SUMMARIZE	Permanent	This document, Section 7.2
BOOK	Permanent	This document, Section 7.2
SCHEDULE	Permanent	This document, Section 7.2
LEARN	Permanent	This document, Section 7.2
DELEGATE	Permanent	This document, Section 7.2
COLLABORATE	Permanent	This document, Section 7.2
CONFIRM	Permanent	This document, Section 7.2
ESCALATE	Permanent	This document, Section 7.2
NOTIFY	Permanent	This document, Section 7.2
DESCRIBE	Permanent	This document, Section 7.2
SUSPEND	Permanent	This document, Section 7.2

Table 27: Initial AGTP Method Registry Entries

9.3. AGTP Status Code Registry

Establishment of a new IANA registry: Agent Transfer Protocol Status Codes.

Registry name: Agent Transfer Protocol Status Codes

Registration procedure: Expert Review + published specification required.

The following AGTP-specific status codes are registered with full definitions:

Code	Name	Definition	Reference
451	Scope Violation	The requested action is outside the Authority-Scope declared in the request headers. The server *MUST* log this event. The agent *MUST NOT* retry the same request without modifying its Authority-Scope declaration. This is a governance signal, not a protocol error.	This document, Section 5.5
452	Budget Exceeded	The requested method execution would exceed the resource limits declared in the Budget-Limit request header. The agent *MUST NOT* retry without modifying the Budget-Limit or reducing request scope. This is a governance signal, not a protocol error. *MUST* be logged.	This document, Section 5.5
453	Zone Violation	The request would route outside the network boundary declared in the AGTP-Zone-ID header. SEP-enforced. The agent *MUST NOT* retry without modifying the AGTP-Zone-ID or obtaining explicit cross-zone authorization. *MUST* be logged.	This document, Section 5.5
550	Delegation Failure	A sub-agent to which a task was delegated via the DELEGATE method failed to complete the task within the declared deadline or returned an error. The response body *SHOULD* contain the sub-agent’s error details.	This document, Section 5.5
551	Authority Chain Broken	One or more entries in the Delegation-Chain header cannot be verified as part of a valid and continuous delegation sequence. The specific unverifiable entry *SHOULD* be identified in the response body.	This document, Section 5.5

		The server *MUST* log this event.	

Table 28: AGTP-Specific Status Code Definitions

9.4. Header Field Registry

AGTP header fields are distinct from HTTP header fields and are registered in a new IANA registry: Agent Transfer Protocol Header Fields.

Registry name: Agent Transfer Protocol Header Fields

Registration procedure: Expert Review + published specification required.

AGTP does not reuse the HTTP Field Name Registry, as AGTP header fields have different semantics, applicability, and versioning constraints from HTTP fields. HTTP header fields are not automatically valid in AGTP, and AGTP header fields are not valid HTTP fields.

Initial registrations (all Permanent): AGTP-Version, AGTP-Method, AGTP-Status, Agent-ID, Principal-ID, Authority-Scope, Session-ID, Task-ID, Delegation-Chain, Priority, TTL, Server-Agent-ID, Attribution-Record, Continuation-Token, Supported-Methods, Budget-Limit, AGTP-Zone-ID, Content-Schema, Telemetry-Export, Cost-Estimate, Attestation-Evidence.

9.5. URI Scheme Registration

Registration of the agtp:// URI scheme per [RFC7595], as described in Section 5.1.8 of this document.

9.6. AGTP Budget Unit Registry

Establishment of a new IANA sub-registry: Agent Transfer Protocol Budget Units.

Registry name: Agent Transfer Protocol Budget Units

Registration procedure: Expert Review per [RFC8126]. New unit registrations *MUST* define: unit name (lowercase ASCII, no spaces or special characters), semantic description, value format (integer or decimal), whether fractional values are permitted, and a reference specification. Units representing financial denominations *MUST* specify the currency and *MUST* define

precision (decimal places). The Designated Expert *SHOULD* verify that the proposed unit does not duplicate an existing registration and that the value format is unambiguous.

Reference: This document

Initial registrations:

Unit	Description	Value Format	Fractional
tokens	Language model token consumption	Integer	No
compute-seconds	CPU/GPU compute time in seconds	Decimal	Yes
USD	US Dollar financial limit	Decimal, 2 places	Yes
EUR	Euro financial limit	Decimal, 2 places	Yes
GBP	Pound Sterling financial limit	Decimal, 2 places	Yes
calls	Downstream API call count	Integer	No

Table 29: Initial AGTP Budget Unit Registry Entries

9.7. Agent Registry Retention Policy

The AGTP registry *MUST* retain records for all registered agents regardless of lifecycle state. The following minimum retention periods apply:

Lifecycle State	Minimum Retention Period
Active	Duration of Active state + 7 years
Suspended	Duration of Suspended state + 7 years
Revoked	10 years from revocation date
Deprecated	7 years from deprecation date

Table 30: AGTP Registry Minimum Retention Periods

The 7-year minimum reflects common enterprise compliance requirements (SOX, GDPR audit trails, HIPAA). Governance platform operators in regulated industries *SHOULD* extend these minimums to match applicable regulatory requirements.

The retained record for a Revoked or Deprecated agent *MUST* include:

* Canonical Agent-ID (permanently retired, not reissued)

* Agent label and org domain at time of registration

* Trust tier at time of registration

* Activation date and activating principal

* Revocation or deprecation date, initiating principal, and reason code

* Genesis audit record hash (pointer to immutable audit log)

* Full Birth Certificate (archived, not publicly accessible)

* All lifecycle state transitions with timestamps

The retained record *MUST NOT* contain package executable contents, active session data, or Authority-Scope grant tokens.

9.7.1. Domain Name Expiry Interaction

If an organization’s org_domain expires or transfers to a new owner:

1. All Active agents registered under the expired domain *MUST* be automatically Suspended within 24 hours of domain expiry detection.

2. The governance platform *MUST* notify the registered principal contact before suspension takes effect, with a minimum notice period of 30 days if domain expiry was predictable.

3. Suspended agents under an expired domain transition to Deprecated state after 90 days if the domain has not been renewed.

4. A new owner of the domain *MUST NOT* inherit prior agent registrations. New ACTIVATE transactions are required.

This policy prevents domain-transfer-based identity hijacking in which an attacker acquires an expired domain to claim the trust history of agents that operated under it.

10. References

10.1. 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>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/rfc/rfc5280>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/rfc/rfc8126>.

[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>.

[RFC8179] Bradner, S. and J. Contreras, "Intellectual Property Rights in IETF Technology", BCP 79, RFC 8179, DOI 10.17487/RFC8179, May 2017, <https://www.rfc-editor.org/rfc/rfc8179>.

[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>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

[RFC9001] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <https://www.rfc-editor.org/rfc/rfc9001>.

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://www.rfc-editor.org/rfc/rfc9334>.

10.2. Informative References

[A2A] Linux Foundation, "Agent-to-Agent Protocol Specification", 2025, <https://a2aprotocol.ai>.

[ACP] IBM Research, "Agent Communication Protocol", 2025.

[AGTP-CERT] Hood, C., "AGTP Agent Certificate Extension", Work in Progress, Internet-Draft, draft-hood-agtp-agent-cert-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-agent-cert-00>.

[AGTP-COMPOSITION] Hood, C., "AGTP Composition with Agent Group Messaging Protocols", Work in Progress, Internet-Draft, draft-hood-agtp-composition-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-composition-00>.

[AGTP-DISCOVER] Hood, C., "AGTP Agent Discovery and Name Service", Work in Progress, Internet-Draft, draft-hood-agtp-discovery-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-discovery-00>.

[AGTP-METHODS]
Hood, C., "AGTP Standard Extended Method Vocabulary", Work in Progress, Internet-Draft, draft-hood-agtp-standard-methods-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-standard-methods-00>.

[AGTP-WEB3]
Hood, C., "AGTP Web3 Bridge Specification", Work in Progress, Internet-Draft, draft-hood-agtp-web3-bridge-00, 2026, <https://datatracker.ietf.org/doc/html/draft-hood-agtp-web3-bridge-00>.

[ANP] "Agent Network Protocol", 2025.

[MCP] Anthropic, "Model Context Protocol", 2024, <https://modelcontextprotocol.io>.

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/rfc/rfc6749>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/rfc/rfc7540>.

[RFC7595] Thaler, D., Ed., Hansen, T., and T. Hardie, "Guidelines and Registration Procedures for URI Schemes", BCP 35, RFC 7595, DOI 10.17487/RFC7595, June 2015, <https://www.rfc-editor.org/rfc/rfc7595>.

[RFC9114] Bishop, M., Ed., "HTTP/3", RFC 9114, DOI 10.17487/RFC9114, June 2022, <https://www.rfc-editor.org/rfc/rfc9114>.

Appendix A. Authority-Scope Format

Authority-Scope values are expressed as a space-separated list of scope tokens following the pattern: [domain]:[action] or [domain]:* for full domain access. Tokens *MUST* be lowercase ASCII with a single colon separator.

Examples:

```
Authority-Scope: calendar:book calendar:query
Authority-Scope: documents:summarize documents:query knowledge:learn
Authority-Scope: *:query
Authority-Scope: booking:* payments:confirm
```

Reserved domains (initial set):

Domain	Description
calendar	Scheduling and time-based resource management
documents	Document access, summarization, and annotation
knowledge	Agent context and memory operations
booking	Reservation and resource allocation
payments	Financial transactions and confirmations
agents	Delegation and collaboration with other agents
escalation	Escalation routing and handler management
activation	Governed agent package activation (ACTIVATE method extension)
discovery	Agent discovery and capability query operations (DISCOVER, DESCRIBE)
budget	Resource budget declaration and QUOTE pre-flight operations
telemetry	Telemetry export and observability operations
zone	Network zone boundary declaration and enforcement
suspend	Session suspension and resumption operations
*	All domains require explicit grant; use with caution

Table 31: Reserved Authority-Scope Domains

Appendix B. Example AGTP Wire Formats

The following examples use a human-readable pseudo-wire format with HTTP-style headers followed by a JSON body. The Content-Type for all AGTP message bodies is application/agtp+json.

B.1. QUERY Request and Response

```
AGTP/1.0 QUERY
Agent-ID: agt-7f3a9c2d
Principal-ID: usr-chris-hood
Authority-Scope: documents:query knowledge:query
Session-ID: sess-a1b2c3d4
Task-ID: task-0042
TTL: 3000
Content-Type: application/agtp+json

{
  "task_id": "task-0042",
  "parameters": {
    "intent": "Key arguments against MCP re: HTTP overhead",
    "scope": ["documents:research", "knowledge:session"],
    "format": "structured",
    "confidence_threshold": 0.75
  }
}

AGTP/1.0 200 OK
Task-ID: task-0042
Server-Agent-ID: srv-knowledge-01
Attribution-Record: [signed attribution token]
Content-Type: application/agtp+json

{
  "status": 200,
  "task_id": "task-0042",
  "result": {
    "results": [{"content": "...", "source": "doc-agtp-research",
                 "confidence": 0.91}],
    "result_count": 1
  }
}
```

B.2. BOOK Request and Response

```
AGTP/1.0 BOOK
Agent-ID: agt-travel-planner
Principal-ID: usr-chris-hood
Authority-Scope: booking:* calendar:book
Session-ID: sess-trip-2026-04
Task-ID: task-0107
Priority: normal
Content-Type: application/agtp+json

{
  "method": "BOOK",
  "task_id": "task-0107",
  "parameters": {
    "resource_id": "flight-AA2847",
    "principal_id": "usr-chris-hood",
    "time_slot": "2026-04-15T08:00:00Z",
    "options": {"seat_preference": "aisle", "class": "economy"},
    "confirm_immediately": true
  }
}

AGTP/1.0 200 OK
Task-ID: task-0107
Attribution-Record: [signed attribution token]
Content-Type: application/agtp+json

{
  "status": 200,
  "task_id": "task-0107",
  "result": {
    "booking_id": "BK-2026-0107",
    "status": "confirmed",
    "resource_id": "flight-AA2847",
    "confirmation_code": "XQRT7Y"
  }
}
```

B.3. ESCALATE Request and Response

```
AGTP/1.0 ESCALATE
Agent-ID: agt-procurement-03
Principal-ID: usr-finance-dept
Authority-Scope: booking:* payments:confirm
Session-ID: sess-procurement-q2
Task-ID: task-0881
Priority: urgent
Content-Type: application/agtp+json

{
  "method": "ESCALATE",
  "task_id": "task-0881",
  "parameters": {
    "task_id": "task-0880",
    "reason": "scope_limit",
    "context": {
      "attempted_action": "BOOK",
      "resource": "vendor-contract-750k",
      "block_reason": "Exceeds agent authorization threshold"
    },
    "recipient": "usr-cfo",
    "deadline": "2026-03-19T09:00:00Z"
  }
}

AGTP/1.0 202 Accepted
Task-ID: task-0881
Server-Agent-ID: srv-escalation-handler
Content-Type: application/agtp+json

{
  "status": 202,
  "task_id": "task-0881",
  "result": {
    "escalation_id": "ESC-0881",
    "routed_to": "usr-cfo",
    "status": "pending_review",
    "task_paused": true,
    "estimated_review_by": "2026-03-19T09:00:00Z"
  }
}
```

Appendix C. Comparison Table

Criterion	AGTP	HTTP/ REST	gRPC	AGMP (MCP, A2A, ...)
Intent-native methods	Yes (12 Tier 1)	No	No	Partial
Intent semantics at protocol level	Native	None	None	Messaging layer only
Built-in agent identity	Yes	No	No	No
Authority scope enforcement	Protocol-level	None	None	Application-layer
Built-in attribution/ audit	Yes	No	No	Varies by impl.
Transport flexibility	TCP/UDP/ QUIC	TCP/TLS	HTTP/2	HTTP
Escalation as first-class primitive	Yes	No	No	No
Ecosystem maturity	Proposed	Mature	Mature	Emerging
Governance/ observability	Native	Manual/ bolt-on	Manual	Limited
Method registry extensibility	Yes (Expert Review)	Frozen (IETF Review)	N/A	N/A
Open core / royalty-free	Yes	Yes	Yes	Yes
Agent Manifest Document	Native (.agtp format)	None	None	None

Tamper-proof identity surface	Yes (hash + signature)	No	No	No	
Browser-accessible agent identity	Yes (read-only)	No	No	No	
URI collision prevention	Domain-anchored	N/A	N/A	N/A	
Agent Birth Certificate	Yes (genesis record)	No	No	No	
Domain-expiry lifecycle handling	Specified	N/A	N/A	N/A	
Capability discovery	Native (DESCRIBE)	None	Reflection (partial)	None	
Resource budget enforcement	Native (Budget-Limit, 452)	None	None	None	
Execution attestation (RATS)	Optional (RFC 9334)	None	None	None	
Observability hooks	Native (Telemetry-Export)	None	None	None	
Network zone enforcement	Native (AGTP-Zone-ID, 453)	None	None	None	
Session suspension/ recovery	Native (SUSPEND method)	None	None	None	
AGMP composition profiles	Normative appendix	N/A	N/A	N/A	

Table 32: AGTP Compared to Existing Approaches

HTTP's method registry (registered with IETF Review per [RFC9110]) is effectively frozen for new semantic methods because any new HTTP method must be backward-compatible with existing HTTP infrastructure globally. AGTP's Expert Review + published spec procedure enables the protocol to evolve its method vocabulary as the agent ecosystem develops, without the backward-compatibility constraints of the HTTP method space.

Appendix D. Glossary

Agent: A software system that executes tasks, makes decisions, and takes actions without continuous human supervision per transaction.

AGMP (Agent Group Messaging Protocol): The collective term for higher-layer AI agent messaging standards that operate over AGTP as their transport substrate, including MCP, A2A, ACP, and ANP. AGMPs define what agents say. AGTP defines how those messages move. See Section 1.6.

Agent Birth Certificate: A cryptographically signed identity document issued to an agent at registration time by a governance platform. The genesis record of the agent's existence; the source from which the canonical Agent-ID is derived. Functions as the agent's social security number: issued once, permanently bound, never reissued. See Section 6.7.

Agent Transfer Protocol (AGTP): The application-layer protocol defined in this document, providing a dedicated transport environment for agent traffic.

Agent-ID: A unique identifier for a specific agent instance, present in all AGTP request headers. In the base spec, derived from the Birth Certificate hash. With [AGTP-CERT], cryptographically bound to a verified identity.

Agent Manifest Document: A signed application/agtp+json document returned when an agtp:// URI is resolved. Derived from the agent's .agent or .nomo package. Contains identity, lifecycle state, trust tier, behavioral scope, and birth certificate fields. Never contains executable content.

AGTP-Zone-ID: A request header declaring the network zone or organizational boundary within which a request must be processed. SEPs *MUST* enforce zone boundaries and return 453 Zone Violation if a DELEGATE or COLLABORATE request would route outside the declared zone.

Attribution Record: A signed, logged record of an agent action, sufficient for audit and compliance purposes. *MAY* include RATS attestation evidence per [RFC9334] for hardware-rooted execution proof in high-stakes domains.

Authority-Scope: A declared set of permissions defining what actions an agent is authorized to take, expressed as space-separated domain:action tokens.

Budget-Limit: A request header declaring the maximum resource consumption the principal authorizes for a method invocation, expressed as space-separated unit=value tokens from the IANA AGTP Budget Unit Registry. Exceeding the declared limit causes 452 Budget Exceeded.

Delegation Chain: An ordered record of Agent-IDs representing the sequence of delegations that led to the current request.

DESCRIBE: An AGTP Tier 1 core method returning the declared capabilities, supported modalities, method vocabulary, and versioned feature set of a specific agent endpoint. Used for pre-task negotiation. Category: ACQUIRE.

ESCALATE: An AGTP method representing an agent’s intentional deferral of a decision or action to a human principal or higher-authority agent. A first-class method, not a failure code.

Governance Token: A signed, time-limited JWT artifact encoding a specific governance verdict for a specific action. The runtime companion to the Birth Certificate. Default TTL: 30 seconds. Must not be reused.

Intent Verb: An AGTP method name expressing the agent’s purpose, as distinguished from HTTP resource-operation verbs (GET, POST, PUT, DELETE).

Method Registry: The IANA-maintained registry of valid AGTP method names and their specifications. Registration requires Expert Review and a published specification.

Principal: The human, organization, or system that authorized an agent to act and is accountable for its actions.

Principal-ID: The identifier of the principal on whose behalf an agent operates, present in all AGTP request headers.

Scope-Enforcement Point (SEP): An AGTP-aware infrastructure

component, load balancer, gateway, proxy, that enforces Authority-Scope and AGTP-Zone-ID compliance on AGTP requests without application-layer access. Requires [AGTP-CERT].

Scope Violation (451): An AGTP status code returned when an agent requests an action outside its declared Authority-Scope. A governance signal, not a protocol error. *MUST* be logged.

Session: An AGTP persistent connection context shared across multiple method invocations within a single agent workflow.

SUSPEND (method): An AGTP Tier 1 core method that places a specific active session into a recoverable paused state, issuing a single-use base64url-encoded 128-bit resumption nonce. Session-scoped; does not affect registry lifecycle state. Category: ORCHESTRATE.

Trust Tier: A classification (1, 2, or 3) assigned to an agent at registration based on the strength of identity verification. Tier 1 requires DNS-anchored domain verification and a .nomo governed package. Tier 2 is org-asserted without DNS verification. Tier 3 is experimental, not production-eligible.

551 Authority Chain Broken: An AGTP status code returned when one or more entries in the Delegation-Chain header cannot be verified as part of a valid and continuous delegation sequence. *MUST* be logged.

Appendix E. AGTP Composition with AGMPs

This appendix provides normative mapping guidance for carrying AGMP messages (MCP, A2A, ACP) over AGTP as their transport substrate. Full composition specifications are provided in [AGTP-COMPOSITION]; this appendix provides the canonical mapping table and precedence rules.

E.1. Precedence Rule

AGTP headers (Agent-ID, Principal-ID, Authority-Scope, Delegation-Chain) take precedence over equivalent fields in the messaging-layer payload for routing, enforcement, and audit purposes. Infrastructure components including SEPs and governance gateways *MUST* use AGTP header values for all protocol-level decisions. Messaging-layer identity fields *MAY* be present in the body for application-layer use but *MUST NOT* override AGTP header values.

E.2. AGMP-to-AGTP Canonical Mapping

AGMP	Concept	AGTP Mapping
A2A	Task	AGTP DELEGATE body; A2A task.id maps to Task-ID header
A2A	Capability	AGTP DESCRIBE response; capability_domains
A2A	Agent Card	AGTP Agent Manifest Document
A2A	Provenance chain	AGTP Delegation-Chain header
A2A	Artifact	AGTP NOTIFY body with content_type: artifact
MCP	Tool call	AGTP QUERY or NOTIFY body
MCP	Context / conversation state	AGTP Session-ID header + LEARN method
MCP	Sampling / inference request	AGTP QUERY with modality: inference
MCP	Resource	AGTP QUERY with appropriate scope
ACP	Agent-to-agent message	AGTP NOTIFY or COLLABORATE body
ACP	Capability advertisement	AGTP DESCRIBE response

Table 33: AGMP-to-AGTP Canonical Mapping

E.3. Wire Example: A2A Task over AGTP

```
AGTP/1.0 DELEGATE
Agent-ID: agtp://agtp.acme.tld/agents/orchestrator
Principal-ID: usr-chris-hood
Authority-Scope: agents:delegate documents:query
Delegation-Chain: agtp://agtp.acme.tld/agents/orchestrator
Session-ID: sess-a1b2c3d4
Task-ID: task-0099
Content-Schema: https://a2aprotocol.ai/schema/task/v1
Content-Type: application/agtp+json

{
  "method": "DELEGATE",
  "task_id": "task-0099",
  "parameters": {
    "target_agent_id": "agtp://agtp.acme.tld/agents/analyst",
    "authority_scope": "documents:query",
    "delegation_token": "[signed token]",
    "task": {
      "a2a_task_id": "a2a-task-7f3a",
      "message": "Summarize Q1 financial reports",
      "artifacts": []
    }
  }
}
```

E.4. Wire Example: MCP Tool Call over AGTP

```
AGTP/1.0 QUERY
Agent-ID: agtp://agtp.acme.tld/agents/assistant
Principal-ID: usr-chris-hood
Authority-Scope: documents:query knowledge:query
Session-ID: sess-mcp-b2c3d4
Task-ID: task-0100
Content-Schema: https://modelcontextprotocol.io/schema/tool-call/v1
Content-Type: application/agtp+json

{
  "method": "QUERY",
  "task_id": "task-0100",
  "parameters": {
    "intent": "web_search",
    "modality": "tool",
    "mcp_tool_name": "web_search",
    "mcp_tool_input": {"query": "IETF agent protocol drafts 2026"}
  }
}
```

Author’s Address

Chris Hood
Nomotic, Inc.
Email: chris@nomotic.ai
URI: https://nomotic.ai