Dispatch Working Group C. Yang
Internet-Draft Beijing University of Posts and Telecommunications
Intended status: Standards Track Z. Liu
Expires: 21 January 2026 Tsinghua University
A. Wang
China Telecom
20 July 2025

# Internet of Agents Protocol (IoA Protocol) for Heterogeneous Agent Collaboration
draft-yang-ioa-protocol-00

## Abstract

This draft defines a new agent collaboration protocol, named the Internet of Agents Protocol (IoA Protocol), to support distributed, heterogeneous agent collaboration in intelligent systems. The IoA Protocol enables dynamic team formation, adaptive task coordination, and structured communication among agents with diverse architectures, tools, and knowledge sources. Through a layered architecture and extensible message format, it supports decentralized deployment across devices and can interoperate with existing frameworks. The protocol is particularly suited to emerging 6G application scenarios such as intelligent transportation, smart healthcare, and large-scale human宴鄭I teaming.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 21 January 2026.

Table of Contents



## 1. Introduction

With the rapid advancement of large language models (LLMs) and multimodal autonomous agents, modern intelligent systems are increasingly constructed as collaborative networks of multiple agents. These agents are expected to work together to solve complex, open-ended tasks. However, they often differ in capabilities, tools, runtime environments, and communication patterns, leading to significant challenges in interoperability, dynamic coordination, and cross-device deployment. As a result, current multi-agent frameworks fall short of the flexibility and generality required in real-world applications.

In a typical collaborative setting shown in Figure 1, agents with specialized functions寰芭ncluding a Google Scholar Agent for academic search, an AI Research Specialist for conceptual planning, a PDF Agent for document analysis, and an Academic Writing Agent for content generation寰芭ust work together to complete a research paper on 寰廬nternet of Agents.寰These agents are distributed across devices (e.g., laptops, edge nodes, cloud services), and each relies on different execution frameworks or data formats.

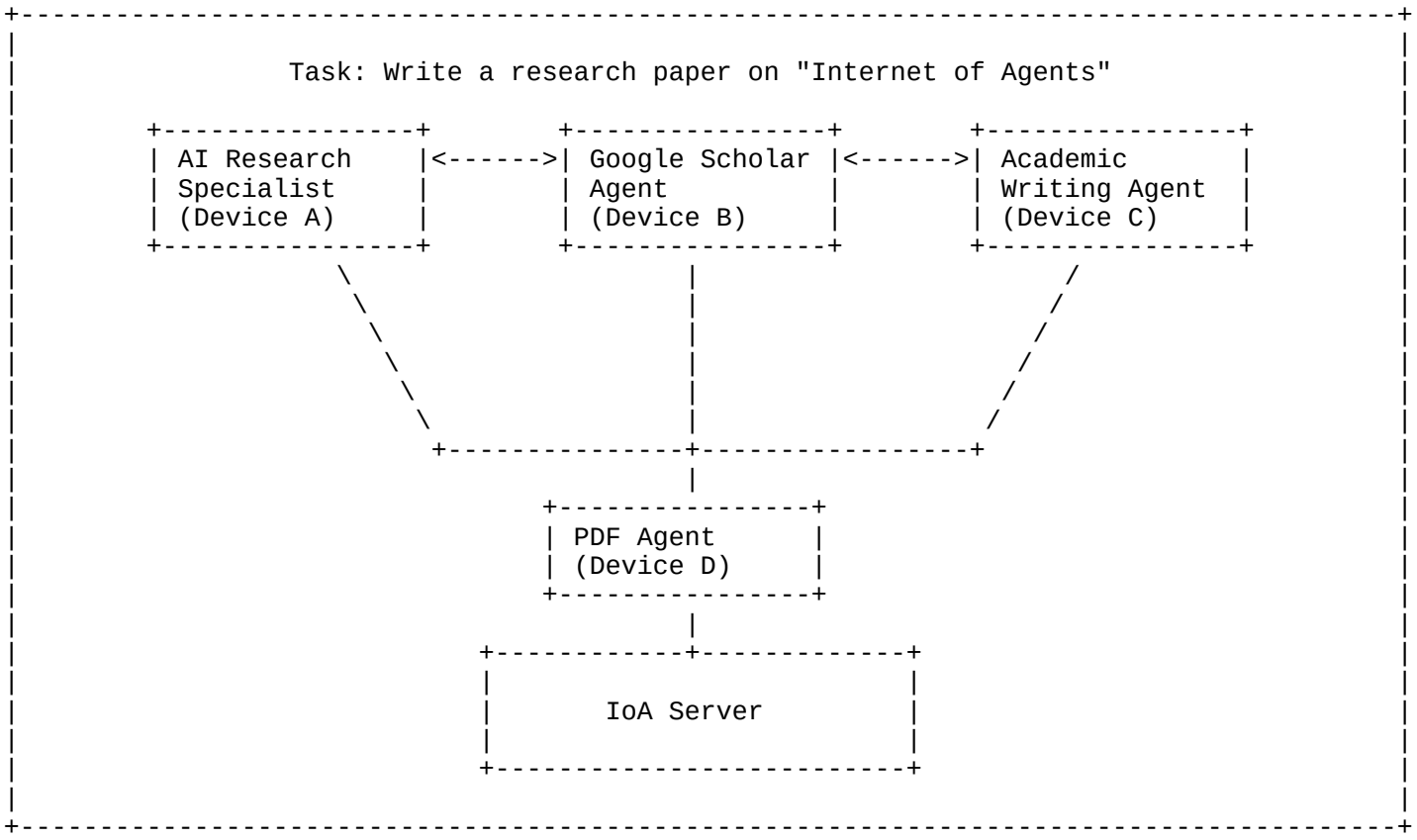


Figure 1: Multi-agent collaboration scenario

When the Google Scholar Agent encounters a specialized PDF parsing task beyond its capability, existing frameworks often fail to dynamically recruit the PDF Agent due to rigid team formation rules. Likewise, when the AI Research Specialist and Writing Agent attempt to synchronize intermediate results in real time, inflexible communication channels may result in delays or dropped information.

Existing solutions exhibit several key limitations:

* Closed frameworks that restrict integration with third-party agents such as AutoGPT or Open Interpreter;

* Single-device simulation that fails to reflect cross-device deployment scenarios typical in edge-cloud collaboration;

* Hard-coded workflows that prevent agents from switching between synchronous and asynchronous task execution at runtime.

To address these challenges, this draft introduces the Internet of Agents (IoA) Protocol—a layered, extensible collaboration standard designed for intelligent multi-agent systems. The core goal of the protocol is to enable seamless collaboration among heterogeneous agents across devices, tools, and execution environments. It supports:

* Agent integration via a standardized interface and registration mechanism;

* Dynamic team formation across distributed environments;

* Finite-state machine-based session control for flexible and autonomous dialogue management;

* Structured message formats with group routing, task assignment, and response coordination.

The design of the IoA Protocol aligns naturally with the vision of 6G networks, which aim to support ubiquitous intelligence through large-scale, low-latency, and semantic-driven communication. By enabling agent collaboration across edge devices, mobile terminals, and cloud nodes, IoA complements 6G's emphasis on edge-cloud-device coordination and distributed AI. Its structured message design, dynamic team formation, and abstracted dialogue control offer the necessary protocol foundation to orchestrate intelligent services over future 6G infrastructures.

## 2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] .

## 3. Terminology

The following terms are defined in this draft:

* IoA: Internet of Agents, a protocol enabling distributed collaboration among heterogeneous agents across devices and 6G networks, defined in Section 4

* Agent Registry Block: A server-side module storing structured capability descriptions of all registered agents, supporting semantic search for team formation, defined in Section 4

* Team Formation Block: A client-side module responsible for initiating, joining, or disbanding agent teams based on task requirements, including nested sub-teams, defined in Section 4

* Session State Machine: A finite-state model governing collaboration states (Discussion, Synchronous Task Assignment, Asynchronous Task Assignment, Pause and Trigger, Conclusion) for adaptive dialogue management, defined in Section 4

* HTTP: Hypertext Transfer Protocol, a application-layer protocol for distributed, collaborative, hypermedia information systems, referenced in IoA for interoperability with web-based agents, defined in [RFC9110]

* JSON-RPC: A remote procedure call protocol encoded in JSON, referenced in IoA for structured communication between web-based agents, defined in [RFC8259]

* QUIC: A transport layer protocol providing secure, low-latency communication over UDP, used in IoA for real-time agent messaging, defined in [RFC9000]

## 4. IOA Methods

### 4.1. IOA Architecture

The Internet of Agents Protocol (IoA Protocol) enables distributed collaboration among heterogeneous agents through a layered architecture and distributed communication protocol. It supports seamless integration across devices, toolchains, and runtime environments.

The IoA system adopts a three-layer architecture implemented symmetrically at both the server and client side:

* Server-side: Handles global coordination, agent discovery, group management, and message routing.

* Client-side: Encapsulates individual agents and provides interfaces for team collaboration and local task execution.

An overview of the layered structure is shown in Figure 2.

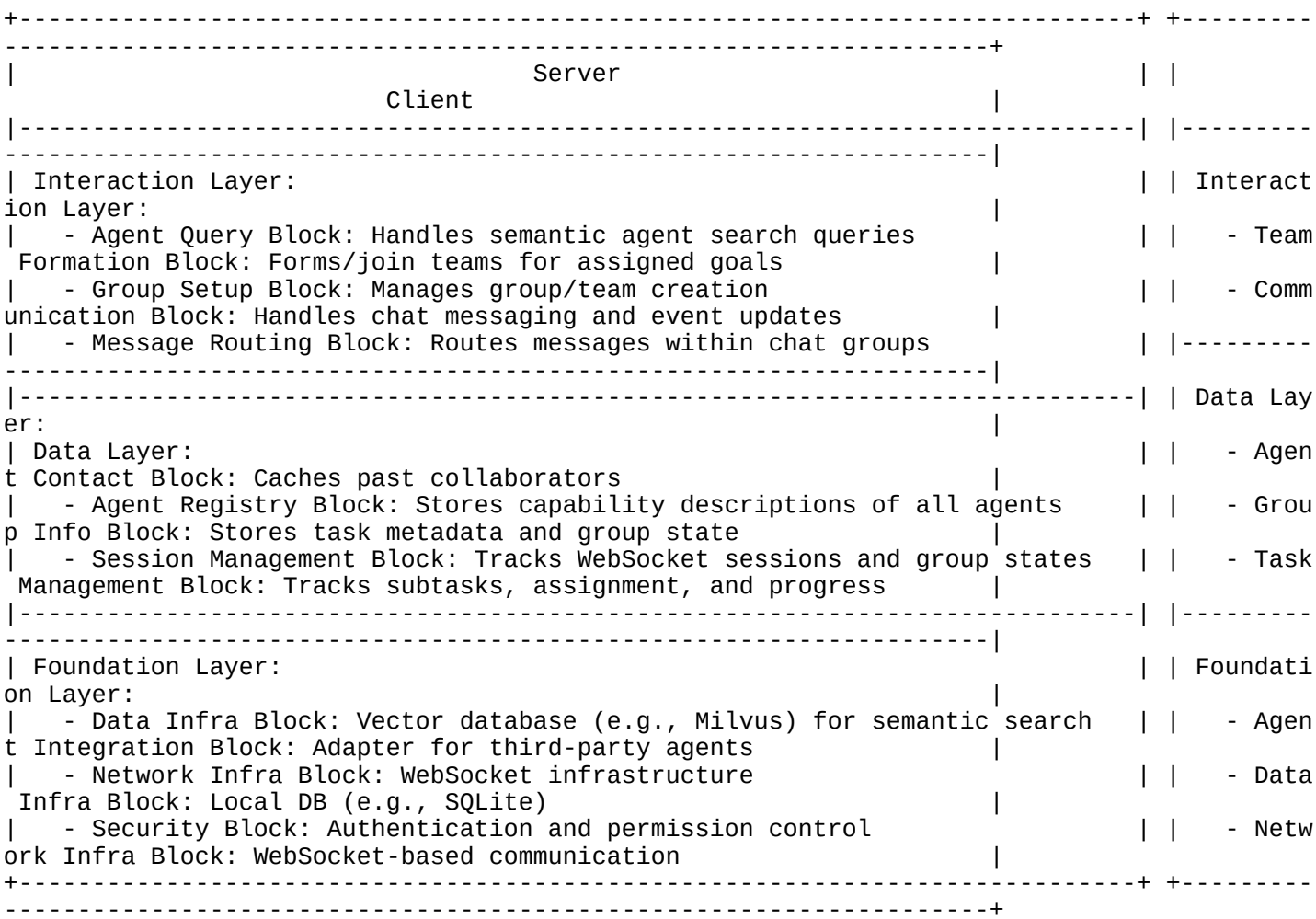


Figure 2: Layered architecture of IoA system

## 4.2. Heterogeneous Agent Integration

IoA supports the integration of heterogeneous agents from diverse sources through a unified interface, including third-party agents such as AutoGPT, Open Interpreter, and embodied robotic agents.

When a new agent joins the IoA, its client wrapper undergoes a registration process with the server. During this registration, the agent is expected to provide a comprehensive description of its capabilities, skills, and domains of expertise. For an agent c_i, its description is denoted as d_i, and is stored in the Agent Registry Block within the Data Layer of the server.

The set of all registered agents is denoted as C = {c, c, ..., c}, where each c_i is associated with its capability description d_i. This mechanism enables future semantic matching and intelligent task allocation.

### 4.3. Autonomous Team Formation

Agents initiate the search process by submitting capability requirements to the Agent Query Block. The server performs semantic matching using vector similarity and returns candidate agents from the Agent Registry Block.

IoA supports nested team structures. An initial group is formed for the main goal, and subgroups are recursively created if subtasks require new capabilities. This forms a hierarchical tree structure, reducing communication complexity and organizational overhead.

The entire team formation process is autonomous, task-driven, device-agnostic, and self-organizing.

### 4.4. Session and Task Management Method

IoA models group conversations and collaboration using a finite-state machine with five abstract states:

* Discussion: Agents engage in general dialogue, exchange ideas, and clarify task require ments;

* Synchronous task assignment: Tasks are assigned to specific agents, pausing the group chat until completion;

* Asynchronous task assignment: Tasks are assigned without interrupting the ongoing discus sion;

* Pause & trigger: The group chat is paused, waiting for the completion of specified asyn chronous tasks;

* Conclusion: Marks the end of the collaboration, prompting a final summary.

State transitions are managed autonomously by a coordinator agent using the conversation history and session context to determine the next state and speaker.

### 4.5. Message Protocol Overview

The agent message protocol in IoA is designed for extensibility and flexibility, enabling effective collaboration among heterogeneous agents. Each message consists of two main parts: a header and a payload.

The header contains essential metadata to ensure proper routing and processing. Key fields include:

* sender: The unique identifier of the agent sending the message.

* group_id: The identifier of the group chat to which the message belongs.

The payload carries the main content of the message and varies depending on message type. Common fields include:

* message_type: Indicates the purpose of the message (e.g., discussion, task assignment, pause and trigger).

* next_speaker: The identifier(s) of the agent(s) expected to respond.

The full structure of the message format is illustrated in Figure 3.

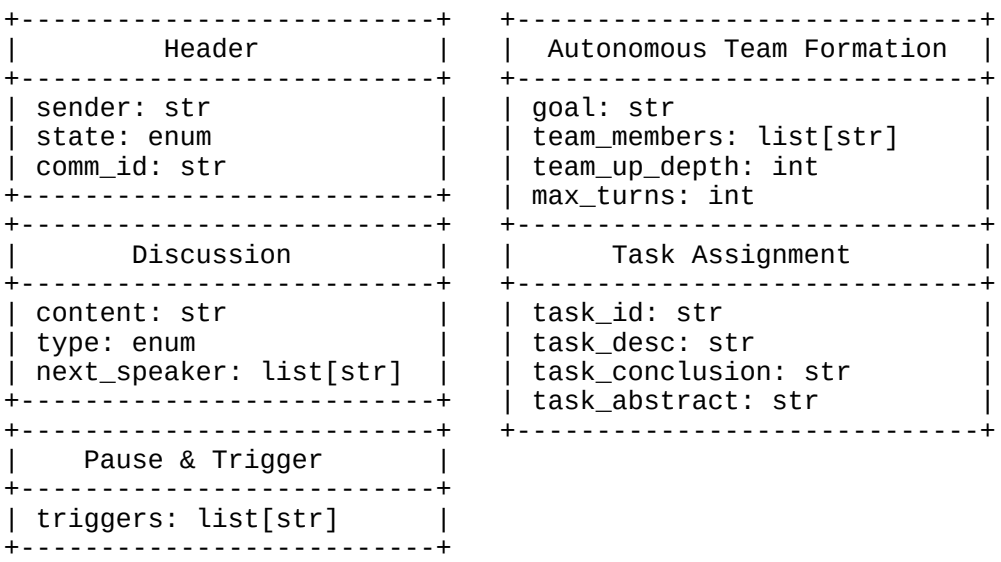


Figure 3: Structure of IoA Message Protocol

## 5. Relation to the A2A Protocol

The Agent-to-Agent (A2A) protocol is a communication standard designed to support standardized, secure, and modality-agnostic interaction between AI agents. Built upon existing web technologies such as HTTP, Server-Sent Events (SSE), and JSON-RPC, A2A emphasizes default security, support for long-running tasks, and cross-modality interoperability. It introduces the concept of an AgentCard to describe agent capabilities, enabling effective discovery and invocation.

The Internet of Agents (IoA) protocol shares the same fundamental goal with A2A: to break down communication barriers among agents and improve the overall efficiency of multi-agent systems. Both protocols rely on network communication technologies and adopt similar approaches to message encoding, decoding, and task coordination.

However, the two protocols diverge significantly in terms of design philosophy and core mechanisms:

* A2A focuses on enabling standardized communication through web-native technologies, effectively creating a "free trade zone" for agents where interoperability is built-in. In contrast, the IoA protocol draws inspiration from Internet architecture and targets the problem of ecosystem fragmentation. It establishes a system-level collaboration platform where heterogeneous agents can freely register, discover one another, and collaborate across platforms and devices.

* A2A is based on HTTP and JSON-RPC for communication, combined with task lifecycle management and capability discovery through AgentCard. IoA, on the other hand, offers a more comprehensive collaboration framework, including agent registration, autonomous nested team formation, finite-state-machine-driven session control, and trigger-based task coordination.

* While A2A is suitable for standardized task responses and streaming updates, it lacks native support for dynamic session management and nested subtask structures. IoA enables adaptive interaction flow via a session state machine, and its team_up_depth field supports recursive team formation and state transitions—making it more effective for handling complex and evolving task scenarios.

In summary, A2A is well-suited for lightweight, standardized task interfaces, whereas IoA provides a more flexible and system-oriented protocol for large-scale, heterogeneous, and dynamic multi-agent collaboration. The two protocols can complement each other at different layers, jointly advancing the development of agent communication technologies.

## 6. Future Enhancements for 6G-Enabled IoA Protocol

To fully realize the potential of 6G-enabled intelligent systems, the Internet of Agents (IoA) protocol requires continuous architectural evolution and standardization. This section outlines key directions for future enhancements to improve scalability, decentralization, interoperability, and network integration.

### 6.1. Distributed Agent Registration and Discovery

The current IoA design relies on a centralized server model, which may limit scalability and introduce single points of failure under large-scale deployment. A promising direction is to adopt a decentralized registration and discovery mechanism, where agents can publish their capabilities to a shared registry accessible via a 6G-compatible web-based interface. Inspired by Domain Name System (DNS) and search engines, agents could be discoverable through keyword-based or semantic search at scale, enabling lightweight browser-based or API-based discovery across domains.

This decentralized lookup layer would allow IoA to support scenarios where agents operate across multiple domains, owners, and physical networks, while still maintaining secure and authenticated interaction through digital signatures and trust mechanisms.

### 6.2. Positioning of the Protocol in the Network Layering System

To achieve efficient integration of the IOA protocol with the 6G network protocol stack, the current design primarily positions IOA at the application layer, built on top of transport and session protocols such as TCP, UDP, WebSocket, and QUIC. From the perspective of functional mapping, the corresponding relationship between IOA’s three-layer architecture and the computer network layers is as follows:

* Interaction Layer → Maps to the application layer, responsible for high-level logic such as message protocols, group collaboration, and session state transitions.

* Data Layer → Spans the application layer and session layer, managing agent states, group metadata, and context tracking.

* Foundation Layer → Corresponds to the transport layer and system infrastructure, including secure communication channels (e.g., WebSocket/QUIC), databases, and network service modules.

Since the IOA protocol involves intelligent behaviors such as agent orchestration, semantic-driven interaction, and session control, an intelligence layer can be introduced above the traditional application layer. This layer encapsulates core intelligent collaboration logic—such as semantic-based agent matching, AI-driven session strategy optimization, dynamic task decomposition, and 6G-aware team reorganization—into standardized message formats. This layer shields upper-layer applications and lower-layer protocols from the complexity of intelligent decision-making, enabling them to focus on their core functions without concerning themselves with the

details of how intelligence is implemented (e.g., scenario-specific task execution at the application layer, reliable data transmission at the transport layer). Its advantages are reflected in: standardizing the collaboration of heterogeneous agents, reducing integration costs across 6G scenarios; improving communication efficiency through semantic compression and 6G feature adaptation; and easily expanding to support new intelligent behaviors and 6G application scenarios through modular updates of the intelligence layer.

### 6.3. Enhanced Scalability and Fault Tolerance

To scale beyond millions of agents, the IoA protocol should adopt sharding and region-based message routing. Distributed registries and dynamic load balancing can reduce latency and avoid bottlenecks. Caching of frequent agent metadata at edge nodes is also critical for fast retrieval in latency-sensitive 6G use cases.

### 6.4. Semantic Interoperability and Ontology Alignment

In highly heterogeneous environments, agents may describe their capabilities using different terminologies. To address this, the IoA protocol should support ontology mapping and alignment mechanisms. This allows agents with differing skill descriptors to still interoperate, using shared or translated task definitions during team formation and dialogue.

### 6.5. Security and Privacy Enhancements

For mission-critical 6G scenarios (e.g., autonomous vehicles, medical AI), the protocol must incorporate stronger security primitives. This includes:

* End-to-end encryption with forward secrecy.
* Support for zero-trust architectures with agent attestation and secure enclaves.
* Fine-grained access control based on agent role and session context.

## 7. Security Considerations

IOA servers and agents store sensitive data including capability descriptors, session state metadata, and task execution logs, which consume memory and computational resources. To mitigate risks of resource exhaustion and unauthorized access, [RFC6749] (OAuth 2.0) mandates that IOA entities must authenticate peers via token-based validation before processing registration requests or collaboration messages. Additionally, all data transmission between entities must use TLS 1.3 as specified in [RFC8446] to ensure confidentiality and integrity, preventing eavesdropping or tampering.

## 8. IANA Considerations

[TBD] This document defines a new protocol for heterogeneous agent collaboration: the Internet of Agents (IoA) Protocol. The protocol's code point allocation will be determined in subsequent revisions as the standard matures, in accordance with IANA's relevant registration procedures.

## 9. Acknowledgement

Thanks Weize Chen, Ziming You, Ran Li, Yitong Guan, Chen Qian, Chenyang Zhao, Ruobing Xie, Maosong Sun and Yu Hao for their valuable comments on this draft.

## 10. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

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

[RFC7432] Sajassi, A., Ed., Aggarwal, R., Bitar, N., Isaac, A., Uttaro, J., Drake, J., and W. Henderickx, "BGP MPLS-Based Ethernet VPN", RFC 7432, DOI 10.17487/RFC7432, February 2015, <https://www.rfc-editor.org/info/rfc7432>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

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

[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/info/rfc9000>.

[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/info/rfc9110>.

## Authors' Addresses

Cheng Yang
Beijing University of Posts and Telecommunications
10 Xitucheng Road, Haidian District
Beijing
Beijing, 100876
China
Email: yangcheng@bupt.edu.cn

Zhiyuan Liu
Tsinghua University
30 Shuangqing Road, Haidian District
Beijing
Beijing, 100084
China
Email: liuzy@tsinghua.edu.cn

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing
Beijing, 102209
China
Email: wangaj3@chinatelecom.cn