Web Authorization Protocol
Internet-Draft
Intended status: Experimental
Expires: 4 January 2026

Y. Song
L. Li
Huawei
F. Liu
Huawei Singapore

3 July 2025

OAuth2.0 Extention for AI Agent: Authorization on Target
draft-song-oauth-ai-agent-authorization-00

Abstract

In this draft, we address to potential adapt authorization frameworks for the future AI agent. An extension is proposed in the OAuth 2.0 protocol with an optional field *target_id* for granular authorization. It can be seen useful for the future virtual/physical AI agents. By explicitly identifying the target during authorization, the draft aims to support precise permission management and enhance traceability. Serveral risks can be mitigated through the proposed extension, such as potential unauthorized or malicious behavior of AI components in the netowrk, while the compatibility of existing OAuth 2.0 workflows is still maintained.

Requirements Language

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 [RFC2119].

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 4 January 2026.

Table of Contents

1. Introduction

The rapid development of current AI agents necessitates urgent research in the field of AI agent security. Specifically, based on the analysis in [I-D.rosenberg-ai-protocols], authorization stands as a key research direction within AI agent security, covering both intra-domain and inter-domain levels. It involves various scenarios such as AI agents accessing APIs (e.g., scenarios under the MCP protocol) and mutual access between AI agents (e.g., scenarios under the A2A protocol). However, AI agents possess the characteristics of autonomy, dynamics, and long-lived identities, making the direct application of procedures defined by existing standards (OAuth) significantly inadequate. What's more, as specified in the EU GDPR, data subjects have the right to know how their data will be used. In the context of authorization, the resource owner, as the provider of data, resources, or services to the AI agent, acts as a data subject. To better comply with regulation, the client should therefore disclose not only its own identity but also the identity of the

specific AI agent executing the actions.

Therefore, it is necessary to adapt OAuth scheme for full-scenario authorization of AI agents. Nevertheless, [I-D.oauth-ai-agents-on-behalf-of-user] has designed a user-delegated authorization mechanism for AI agents based on "actor" to implement user-delegated authorization for AI agents. It is observed in [I-D.yao-agent-auth-considerations] that AI agents can be devided into two categories: physical AI agent and vitual AI agent. This draft will consider the scenario where the AI agent itself acts as a client, which includes the following two use cases based on the category.

* Use Case 1: The user deploys an AI model on a client that acts as a virtual AI agent. In this case, the AI agent is a module on the client. For example, the client could be a smartphone, where the user downloads a ChatGPT model and instantiates an AI agent locally.

* Use Case 2: The client is a physical AI agent equipped with multiple AI models differing in functionality, versions, or vendors. In this case, the AI model is a module on the AI agent. For example, if developers provide only an AI agent framework with APIs and functional blocks, the user can select Model A for chatting and Model B for confidential work within their company, then install both into the agent.

As previously noted, AI agents function differently from other modules. Once authorized by the user, an AI agent can make independent decisions and execute tasks automatically. While this brings convenience to daily life, it also introduces significant uncertainty due to unexplainable problem of AI. Specifically, an AI agent or AI model could be compromised or trained to perform harmful tasks even as other client modules remain unaffected -- this risk is heightened when the client and the AI agent or AI model originate from different providers. In such cases, failing to identify the specific AI agent or model involved would hinder root cause analysis. Moreover, authorizing at the client level might lead to accidental denial of authorization on the client, disrupting the client’s normal operation.

Therefore, for these use cases, authorization needs to clarify which specific module of the client is being authorized. This draft proposes an authorization mechanism centered on a *target* -- role introduced to identify the client module requiring authorization. To support this, an optional extension field named *target_id* is added to the OAuth 2.0 protocol flow. The *target* may refer to virtual AI agents deployed on the client or AI models hosted on a physical AI agent.

This solution clearly distinguishes between the client and its AI agents or models, preventing unintended disruptions. Additionally, it allows users to select the appropriate AI agent or AI model on the client to grant consent to. For example, in daily scenarios, they may authorize Model A, while in work scenarios, they may choose Model B in previous example. Furthermore, users can clearly see which AI agent or model is requesting permission, enabling better management of local AI components. For instance, if Model B requests access to the user’s digital album, the user can reject the request and even revoke Model B’s permission to access all files on a specific disk.

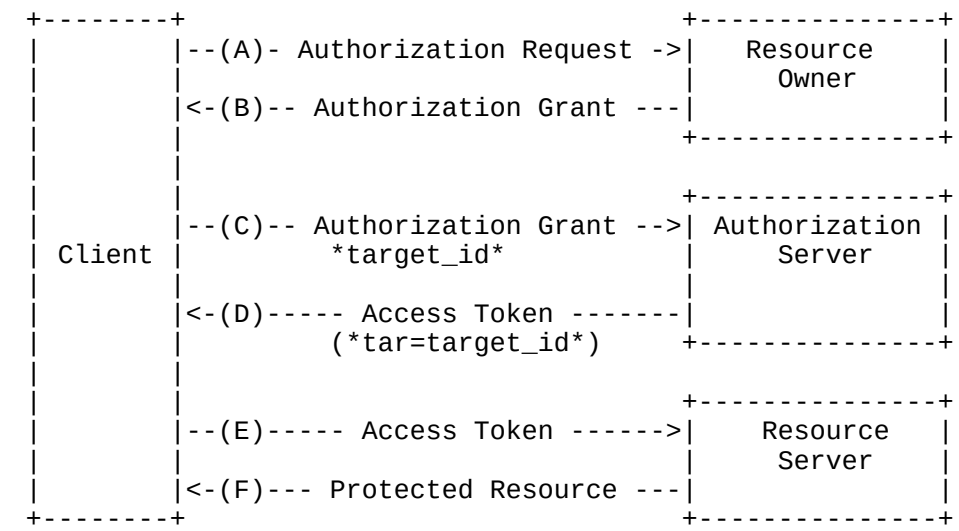
2. Terminology

Target: a module on the client that actually utilize the resource provided by resource owner, e.g. a virtual AI agent deployed on the client, or an AI model deployed on the client (physical AI agent).

This draft uses the terms "authorization server", "client", "resource server" defined by OAuth 2.0 [RFC6749]

3. Extention of Protocol Flow

This protocol flow is modeled after [RFC6749] and extends it with a *target_id* field to enable authorization for the target.



3.1. Extension for Client Registration

Before initiating the protocol, the client registers with the authorization server, as defined in [RFC6749].

In this draft, information about the AI agent or AI model is included as part of the client registration process, including details such as identity and capabilities. For instance, when a client registers using a profile, if it is deployed with an AI agent or AI model, the profile will incorporate relevant information about them. Additionally, the client itself maintains awareness of the AI agent or AI model information.

3.2. Authorization Request & Response

At first the client requests authorization from the resource owner and receives an authorization grant, as defined in [RFC6749].

When the client needs to execute a task, it selects the AI agent or AI model capable of performing the task. Selection methods may include:

* Semantic matching based on the capabilities of the AI agent or AI model and the task description. The task description may be received from other entities or determined by the client. For example, if a model’s capability is "image recognition" and the task description is "identify geographical location from an image", a match is established.

* Selecting an available AI agent or model based on operational status. For example, both Model A and Model B support image recognition, but Model A is in use while Model B is idle, so Model B is chosen.

* Choosing the appropriate AI agent or AI model based on the availability of computing resources and the resource consumption of each option.

* The client acts as an "entry point" to uniformly receive user requirements and then internally selects models according to preset rules. For instance, after the user authorizes the "office scenario," the client automatically selects Model B.

Following this step, the chosen AI agent or AI model is designated as the *target*.

3.3. Access Token Request & Response

The client requests an access token by authenticating with the authorization server and presenting the authorization grant, as defined in [RFC6749], and the message includes the following parameters:

target_id

: OPTIONAL. The identifier of the target which is determined to utilize the requested resource.

code

: REQUIRED. The authorization code received from the authorization server, as defined in [RFC6749].

client_id

: REQUIRED, if the client is not authenticating with the authorization server, as defined in [RFC6749].

The authorization server authenticates the client and validates the authorization grant, as defined in [RFC6749]. Then, the authorization server further validates whether the target identified by the target_id in the message is successfully registered and whether the target is bound to the client. For example, the client’s profile includes the target_id. If valid, the authorization server issues an access token containing a claim indicating that the target is authorized.

If the validation is successful, the access token response message includes an access token. The acess token is similar to OAuth2.0, except that it consists of an additional claim to specify the target.

tar

OPTIONAL. The identifier of the target which is authorized to utilize the requested resource.

If the validation fails, the access token response message includes the reason for failure. [RFC6749] has defined types of error response. In this case, if the target validation fails, the authorization server may reuse the unauthorized_client message and indicate that the target is not registered.

The client requests the protected resource from the resource server and authenticates by presenting the access token, as defined in [RFC6749]. Additionally, it should send the target_id to the resource server to indicate that the requested resource will be used by the target. The resource server validates the access token, as defined in [RFC6749] and also validates whether the received target_id is included in the access token. If the validation succeeds, the resource server provides the resource to the client, which then passes the resource to the AI agent or AI model identified by the target_id.

4. Security Considerations

TBD

5. IANA Considerations

TBD.

6. References

6.1. References

6.2. 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>.

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

6.3. Informative References

[I-D.oauth-ai-agents-on-behalf-of-user]
Senarath, T. S. and A. Dissanayaka, "Further considerations on AI Agent Authentication and Authorization Based on OAuth 2.0 Extension", Work in Progress, Internet-Draft, draft-oauth-ai-agents-on-behalf-of-user-01, 8 May 2025, <https://datatracker.ietf.org/doc/html/draft-oauth-ai-agents-on-behalf-of-user-01>.

[I-D.rosenberg-ai-protocols]
Rosenberg, J. and C. F. Jennings, "Framework, Use Cases and Requirements for AI Agent Protocols", Work in Progress, Internet-Draft, draft-rosenberg-ai-protocols-00, 5 May 2025, <https://datatracker.ietf.org/doc/html/draft-rosenberg-ai-protocols-00>.

[I-D.yao-agent-auth-considerations]
Yao, K., "Further considerations on AI Agent Authentication and Authorization Based on OAuth 2.0 Extension", Work in Progress, Internet-Draft, draft-yao-agent-auth-considerations-00, 30 June 2025, <https://datatracker.ietf.org/doc/html/draft-yao-agent-auth-considerations-00>.

Authors' Addresses

Yurong Song
Huawei
Email: songyurong1@huawei.com

Lun Li
Huawei
Email: lilun20@huawei.com

Faye Liu
Huawei Singapore
Email: liufei19@huawei.com