Title: Cross-Domain AuthZ Information sharing for Agents

Commenté [F(1]: Choose a title that focuses on the problem being solved

Commenté [J(2R1]: Nice title

Abstract

Distributed Multi-Agent Systems consist of Agents and MCP Servers operating across multiple administrative domains, each with its own Identity Providers (IdPs) and Authorization Servers (AS). This document discusses the challenges and solution approaches for sharing authorization information securely and flexibly across domains, including the use of dynamic identity, interoperable claims, and verifiable credentials.

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."

Copyright Notice

Table of Contents

1. Introduction

Distributed Multi-Agent Systems contain Agents and MCP Servers distributed across multiple administrative domains. These domains often use different Identity Providers (IdPs) and Authorization Servers (AS), presenting significant challenges for secure, interoperable sharing of authorization information.

2. Abbreviations / Definitions

- **Agent**: an autonomous entity that performs tasks on behalf of a user or another program within a multi-agent system.
- **MCP Server**: a program that exposes specific capabilities, like data access or tools, to AI models through the Model Context Protocol (MCP).
- **AS** (Authorization Server): a server that issues authorization tokens to clients after successfully authenticating them.
- **IdP** (Identity Provider): is a service that creates, manages, and verifies digital identities for users and service users, often for accessing multiple applications and services.
- **Subject**: the **principal** (user or service),
- **Claim**: is essentially a piece of information asserted about a **subject,**
- **Domain**: scope over which a specific, uniform set of security policies and user management rules are enforced.

3. Problem Statement

Distributed Multi-Agent Systems contain Agents and MCP Servers that are distributed across multiple administrative domains. These systems involve different Identity Providers (IdPs) / Authorization Servers (AS) posing challenges in authorization information sharing across these entities.

4. Use Cases

Predictive Threat Detection

Description: Enhance security by predicting threats based on behavior analysis.

Use Case: An organization uses behavior analysis systems to analyze agent behavior patterns. Client-ID Metadata provides a framework for registering agents, while W3C VCs ensure identity verification and provide additional context for the intended tasks of a specific agent (Ex: may access production logs but not PII). The system predicts potential security breaches by assessing deviations in agent behavior and recommends preventive actions, aligning with Zero Trust strategies.

Automated Compliance Monitoring

Description: Automate compliance checks across distributed systems.

Use Case: A financial institution monitors compliance with regulatory requirements. Agents registered through Client-ID Metadata are verified for compliance based on their provided proofs as W3C VCs (Ex: this agent uses this model approved by policy X). The system automatically evaluates compliance status based on provided VCs and generates alerts for any discrepancies, ensuring adherence to Zero Trust policies.

5. Requirements

A system for cross-domain sharing of authorization information of agents needs to meet the following set of requirements:

- **Dynamic Identity for Agents and MCP Servers**: The Agents and MCP Servers can onboard dynamically and get an assigned identity in the IdP.
- **Interoperability Across Domains**: The system must enable seamless interaction between Agents across different domains.
- **Flexible Definition of Claims**: Claims should be adaptable as they may vary from organization to organization.
- **Dynamic Management of Authorization Information**: The system must allow creation, removal, updating, and deletion of authorization information for agents, as agents can be highly dynamic.
- **Security and Privacy**: Ensuring secure and private sharing of authorization information across domains.
- **Compliance and Auditability**: Support for compliance with regulatory standards and auditability of authorization exchanges.

Solution Approach 1: Dynamic Client Registration (RFC 7591)

Description: Dynamic Client Registration as outlined in RFC 7591 allows clients to register with an AS dynamically, facilitating the management of client credentials and metadata.

Discussion:

Commenté [F(3]: Consider doing this "discussion" in table-format. That might be easier for the reader to digest and also look less strange. Statements like <requirements not met: all the rest> looks a bit odd to me TBH.

Requirement	Met or Partially Met	Not Met
Dynamic Identity for Agents and MCP Servers	**Met**	
Interoperability Across Domains	**Partially Met** (Granted by the Dynamic Registration, however no common ground between AS)	
Flexible Definition of Claims		**Not Met** (OAuth2 claims only)
Dynamic Management of Authorization Information		**Not Met** (Delete or Update not supported)
Security and Privacy		**Not Met** (Claims are shared but no common schema)
Compliance and Auditability	**Partially Met** (Claims are shared, APIs are auditable, but no regulatory standards are defined for the schema)	

Solution Approach 2: SCIM extensions

- **draft-abbey-scim-agent-extension**: proposes extending the SCIM (System for Cross-Domain Identity Management) protocol to standardize the provisioning, management, and governance of AI agents and other digital workers.
- **draft-wahl-scim-agent-schema**: companion document to the draft-abbey-scim-agent-extension. Its purpose is to define the specific SCIM schema and attributes required to represent AI agents (digital workers) and agentic applications within the SCIM framework.

Description: Provision Agents / MCP Servers through a SCIM extension.

Discussion:

Requirement	Met or Partially Met	Not Met
Dynamic Identity for Agents and MCP Servers	**Partially Met** (SCIM support is limited and with constraints, example for Azure: "Subsequent syncs are triggered every 20-40 minutes".)	
Interoperability Across Domains	**Partially Met** (SCIM is supported and schema is well defined)	
Flexible Definition of Claims	**Met** (Defined by the schema)	
Dynamic Management of Authorization Information	**Partially Met** (SCIM support is limited and with constraints, example for Azure: "Subsequent syncs are triggered every 20-40 minutes".)	
Security and Privacy	**Partially Met** (A schema is defined but Selective Disclosure not supported)	
Compliance and Auditability	**Partially Met** (Claims are shared, APIs are auditable, but no regulatory standards are defined for the schema)	

Commenté [F(4]: Could you summarize what these drafts state. It would be good if the doc is self-contained, rather than you ask the reader to read through other docs...

Commenté [F(5]: See above. A table would be great. Especially for the requirements that are not met: Explain why they are not met.

Solution Approach 3: Client-ID Metadata

Description: Client ID Metadata (https://datatracker.ietf.org/doc/draft-ietf-oauth-client-id-metadata-document) is a method for OAuth clients to identify themselves using a URL, which points to a JSON file containing their OAuth metadata. Instead of requiring pre-registration, an authorization server can fetch this JSON document at the client's provided URL to provision the identity.

Discussion:

Requirement	Met or Partially Met	Not Met
Dynamic Identity for Agents and MCP Servers	**Met**	
Interoperability Across Domains	**Met**	
Flexible Definition of Claims		**Not Met*** (client_id is common, all the rest is limited to OAuth2 claims)
Dynamic Management of Authorization Information	**Met**	
Security and Privacy	**Partially Met**** (A schema is defined but Selective Disclosure not supported)	
Compliance and Auditability	**Partially Met**** (Claims are shared, APIs are auditable, but no regulatory standards are defined for the schema)	

***Flexible Definition of Claims**

The OAuth claims in the draft follow the claims exposed and defined by:

https://www.iana.org/assignments/oauth-parameters/oauth-parameters.xhtml#client-metadata.

Hence, the metadata fields are limited to a fixed set of claims. This makes it difficult to convey richer features about the client, including provenance details, compliance attestations, or contextual trust scores without introducing vendor-specific extensions, thereby reducing portability and interoperability.

Security and Privacy, **Compliance and Auditability**

All metadata is published to a publicly accessible endpoint. There is no native ability to keep certain attributes private, apply selective disclosure, or restrict visibility based on trust relationships. Sensitive operational data or compliance-related data must be handled outside the metadata framework, leading to fragmented trust models. There is no mechanism for revocation and expiration.

Solution Approach 4: Client-ID Metadata + W3C Verifiable Credentials Extension

Description: The proposed solution is based on Solution 3, but enhances the supported claims with a new claim, `"vc+jwt"`, based on the Verifiable Credentials (https://www.w3.org/TR/vc-data-model-2.0/) from (W3C):

vc+jwt	a standard method used in decentralized identity to **package and secure a digital credential**. The **Verifiable Credential (VC)**—the actual data proving a fact (e.g., identity or degree)—is encapsulated within a **JSON Web Token (JWT)**, which provides a **cryptographic signature** from the issuer. This signature allows any third party (the Verifier) to instantly confirm the credential's **authenticity and integrity**. Modern formats like **SD-JWT VC** further enhance this by enabling the holder to share only specific parts of the credential (Selective Disclosure) to protect privacy.

Discussion:

Commenté [F(6]: Table format?

Requirement	Met or Partially Met	Not Met
Dynamic Identity for Agents and MCP Servers	**Met**	
Interoperability Across Domains	**Met (1)**	
Flexible Definition of Claims	**Met (2)**	
Dynamic Management of Authorization Information	**Met**	
Security and Privacy	**Met (3)**	
Compliance and Auditability	**Met (4)**	

Detailed explanations:

The **W3C Verifiable Credentials Data Model 2.0** addresses the shortcomings (*, **) of the previous Solution 3, by introducing a cryptographically verifiable and standards-based way to assert identities, capabilities, and compliance attestations. It supports selective disclosure to protect sensitive information, enables time-bound and revocable authorizations, and ensures provenance tracking for audit and regulatory compliance. They are part of a broader decentralized identity (DID) specification.

Commenté [F(7]: Which shortcomings?

Using the W3C Verifiable Credentials Data Model 2.0 for **Agents, MCP Clients,** and **MCP Servers** provides a standardized, cryptographically secure way to establish trust, delegate authority, and ensure interoperability across different systems. It enables Agents, MCP Clients, and MCP Servers to prove, among other things, their identities, identity issuers and provenance, supported skills and permitted actions through verifiable proofs, including support for fine-grained and time-bound access control, while maintaining transparency enabling audit trails for compliance. Additionally, it

enhances privacy through selective disclosure and privacy-preserving proofs (e.g., allowing Agents to prove skills and authorized actions without exposing unnecessary details).

The Verifiable Credentials Data Model 2.0 is a W3C standard that fits the needs expressed above:

Commenté [F(8]: Why?

1) Enhanced Interoperability

VCDM 2.0 provides a standardized, extensible way to express identities, capabilities, and compliance data across different systems. Its use of DIDs and schema-governed claims ensures that Agents, MCP Clients, and MCP Servers can verify and exchange trust information, enabling consistent interoperability across domains and platforms.

2) Flexible Definition of Claims

VCDM 2.0 allows arbitrary, schema-governed claims that can represent complex concepts such as provenance, software supply-chain attestations (e.g., SLSA), security posture, audit certifications (SOC2, ISO-27001), or dynamic trust signals. These claims can be extended without breaking interoperability, thanks to widely adopted JSON-LD and schema registries.

3) Security and Privacy

Credentials are signed using verifiable cryptographic proofs (e.g., JSON Web Signatures, Data Integrity proofs). This ensures that Agents, MCP clients, and MCP servers can verify the authenticity of claims without relying solely on HTTPS endpoints or centralized registries.

Through techniques like BBS+ signatures, holders can disclose only the subset of claims required for a given authorization flow. This is essential for scenarios where Agents must prove capabilities (e.g., "I'm allowed to execute this skill") without revealing unnecessary or sensitive identity attributes.

4) Compliance and Auditability

VCDM 2.0 enables cryptographically verifiable audit trails by binding each credential to an issuer, timestamp, and revocation status. Compliance attestations can be embedded directly in credentials, and real-time revocation checks ensure they remain valid. This provides trustworthy provenance, regulatory-grade accountability, and reliable lifecycle tracking across distributed systems.

6. Example using Outshift Identity Service + Okta (IdP)

Metadata example

```
{
  "client_id": "https://identity_service/mcp-001/oauth/client-
metadata.json",
  "client_name": "Example CIMD Client",
  "jwks_uri": "https://identity-service/XYZ/.well-known/jwks.json",
  "token_endpoint_auth_method": "private_key_jwt",
  "vc+jwt": "{VC with JOSE envelope}",
}
```

VC example

```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://example.org/mcp-server/schema/v1"
  ],
  "type": ["VerifiableCredential", "McpServerCredential"],
  "issuer": "did:web:XYZ.okta.com",
  "jwks_uri": "https://identity-service/XYX/.well-known/jwks.json",
  "issuanceDate": "2025-11-10T00:00:00Z",
  "credentialSubject": {
    "id": "https://identity_service/mcp-001/oauth/client-metadata.json",
    "name": "MCP-001",
    "version": "3.2.1",
    "compliance": ["ISO-27001", "GDPR"],
    "trustScore": 0.94
  },
  "proof": {
    "type": "Ed25519Signature2020",
    "created": "2025-11-10T00:00:00Z",
    "proofPurpose": "assertionMethod",
    "verificationMethod": "did:web:identity-service/XYZ#keys-1",
    "jws": "eyJhbGciOiJFZERTQSJ9..xyzsignature"
  }
```