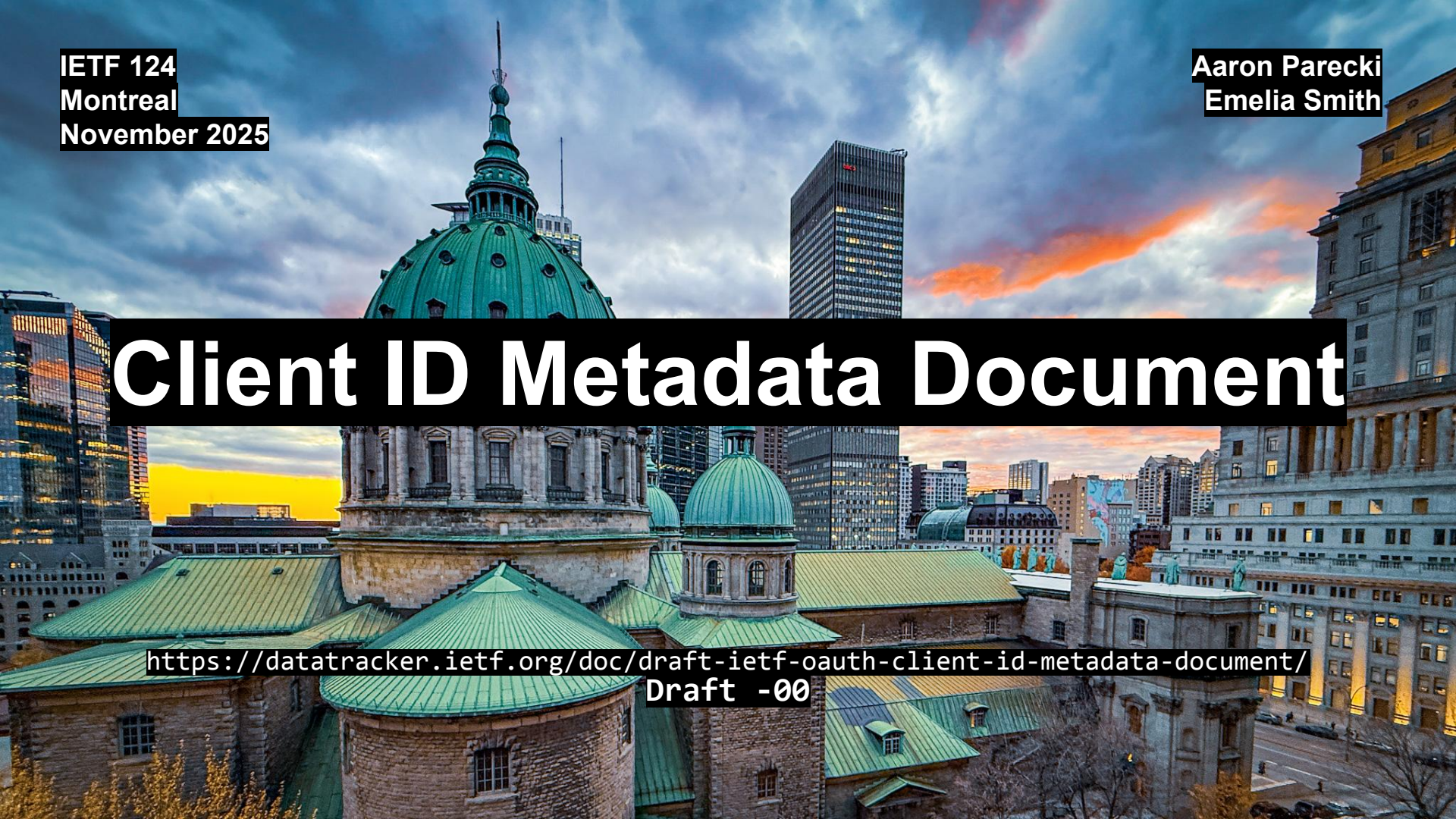

IETF 124
Montreal
November 2025
Aaron Parecki
Emelia Smith
Client ID Metadata Document
https://datatracker.ietf.org/doc/draft-ietf-oauth-client-id-metadata-document/
Draft -00

What: Client ID Metadata Document

- The client publishes their metadata at a URL
  (using the Dynamic Client Registration vocabulary)
  - https://www.iana.org/assignments/oauth-parameters/oauth-parameters.xhtml#client-metadata
  - Doesn't have to be at a predefined path, but it should be a "stable URL", and may be displayed to the end user
- This URL is used as the Client ID in the authorization request
  - `…&client_id=https://example.com/client.json&scope=…`

https://datatracker.ietf.org/doc/draft-parecki-oauth-client-id-metadata-document/

What: Client ID Metadata Document

The client publishes their metadata at a URL (using the Dynamic Client Registration vocabulary)

```
{
  "client_id": "https://example.com/client.json",
  "client_name": "Example Client",
  "client_uri": "https://example.com",
  "logo_uri": "https://example.com/logo.png",
  "redirect_uris": [
    "https://example.com/redirect"
  ]
}
```

The URL is used in an Authorization request

`/authorize?client_id=https://example.com/client.json&...`

https://datatracker.ietf.org/doc/draft-parecki-oauth-client-id-metadata-document/

Why?

Pre-registration of clients is not possible when the client developer has no prior relationship with the authorization server, for example:

- Open source chat app connecting to self-hosted chat server
- Apps that connect to self-hosted services, such as Mastodon or WordPress, where there is no single central server to register the OAuth client
- “Federations” where entities can join through a trust chain but are not all pre-registered with each other.
- MCP Clients that allow connections to arbitrary MCP servers

What’s new since IETF 123?

OAuth Client ID Metadata Document

draft-ietf-oauth-client-id-metadata-document-00

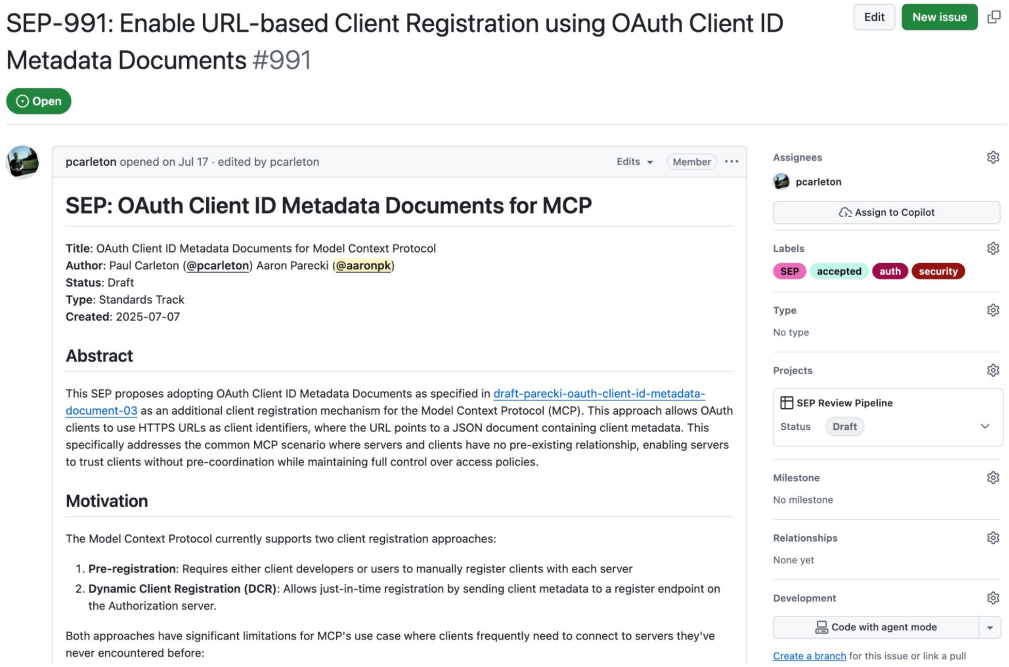
Status | IESG evaluation record | IESG writeups | Email expansions | History

Versions:

00

Adopted!

MCP (Model Context Protocol) is adopting Client ID Metadata Documents as an alternative to Dynamic Client Registration



SEP-991: Enable URL-based Client Registration using OAuth Client ID Metadata Documents #991

Edit | New issue

Open

pcarleton opened on Jul 17 · edited by pcarleton — Edits | Member

SEP: OAuth Client ID Metadata Documents for MCP

Title: OAuth Client ID Metadata Documents for Model Context Protocol
Author: Paul Carleton (@pcarleton) Aaron Parecki (@aaronpk)
Status: Draft
Type: Standards Track
Created: 2025-07-07

Abstract

This SEP proposes adopting OAuth Client ID Metadata Documents as specified in draft-parecki-oauth-client-id-metadata-document-03 as an additional client registration mechanism for the Model Context Protocol (MCP). This approach allows OAuth clients to use HTTPS URLs as client identifiers, where the URL points to a JSON document containing client metadata. This specifically addresses the common MCP scenario where servers and clients have no pre-existing relationship, enabling servers to trust clients without pre-coordination while maintaining full control over access policies.

Motivation

The Model Context Protocol currently supports two client registration approaches:

1. **Pre-registration**: Requires either client developers or users to manually register clients with each server
2. **Dynamic Client Registration (DCR)**: Allows just-in-time registration by sending client metadata to a register endpoint on the Authorization server.

Both approaches have significant limitations for MCP's use case where clients frequently need to connect to servers they've never encountered before:

Assignees: pcarleton
Assign to Copilot

Labels: SEP, accepted, auth, security

Type: No type

Projects: SEP Review Pipeline — Status: Draft

Milestone: No milestone

Relationships: None yet

Development: Code with agent mode

Create a branch for this issue or link a pull

Planned to launch in the November 25, 2025 MCP spec update
https://github.com/modelcontextprotocol/modelcontextprotocol/issues/991

Client ID Metadata Document Use Cases

Client ID Metadata Document without automatic fetching

- An AS can optionally require that Client ID Metadata Document URLs are pre-registered
- This enables enterprise use, enabling the enterprise IdP to allow/disallow certain `client_id` URLs and/or domains, **without the client doing anything special**

Preventing client impersonation

- **Web server clients:**
  - Publish `jwks_uri` and use RFC7523
- **iOS/Android clients:**
  - Use “Attestation Based Client Authentication” and put keys in the “Client Attester Backend”
  - Use app-claimed `https` URL patterns
- **Desktop clients:**
  - No standard or standards-adjacent pattern that I know of
  - Push a credential via MDM linked to Client Attester Backend?
  - Convince desktop OSs to implement app attestation like mobile?

Client ID Metadata Document with JWKS

- A client that can publish public keys to the `jwks_uri` can then use the private key as RFC7523 Client Authentication (Section 2.2 of RFC7523)
- A client that doesn't/can't authenticate to an AS would not include a `jwks_uri`
- *But every client will still include its `client_id` URL*

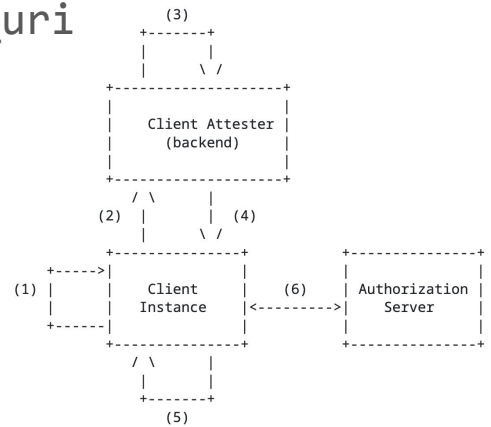
```
{
  "client_id": "https://example.com/client.json",
  "client_name": "Example Client",
  "client_uri": "https://example.com",
  "logo_uri": "https://example.com/logo.png",
  "redirect_uris": [
    "https://example.com/redirect"
  ],
  "jwks_uri": "https://example.com/keys.json"
}
```

Client ID Metadata Document for Native Apps

- A native app can’t host a document at a public URL
- iOS and Android provide platform attestation services, which can be leveraged by “Attestation-Based Client Authentication”
- The “Client Attester Backend” can hold the private keys published at the `jwks_uri`
- The app developer can publish the Client ID Metadata Document on the app’s website, linking to the Client Attester Backend’s `jwks_uri`

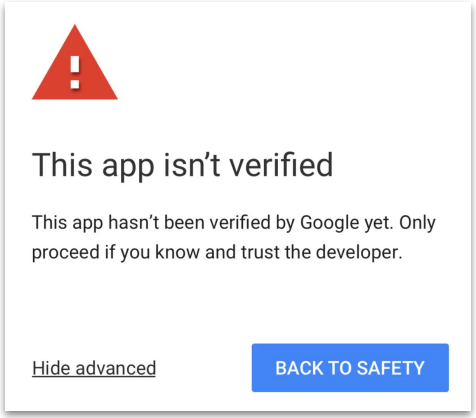
This makes client impersonation of native apps unreasonably expensive.

Should this be specified explicitly in the draft?



https://datatracker.ietf.org/doc/html/draft-ietf-oauth-attestation-based-client-auth#section-1

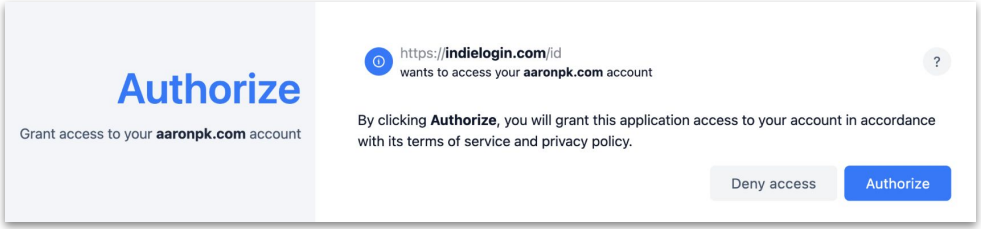
Client ID Reputation



An AS can incrementally trust more metadata properties over time

- The first time a `client_id` URL is seen:
  - show only domain name
  - require matching domain on `redirect_uri`
  - show extra bright red consent screen
- After ~10 users authorize, remove giant warning
- After ~100 users authorize, fetch the metadata and display client name
- After ~1000 users authorize, display client logo
- After manual review of the app, add hyperlink on consent screen to `client_uri`

All of this is up to the AS to decide without requiring different behavior of the client.

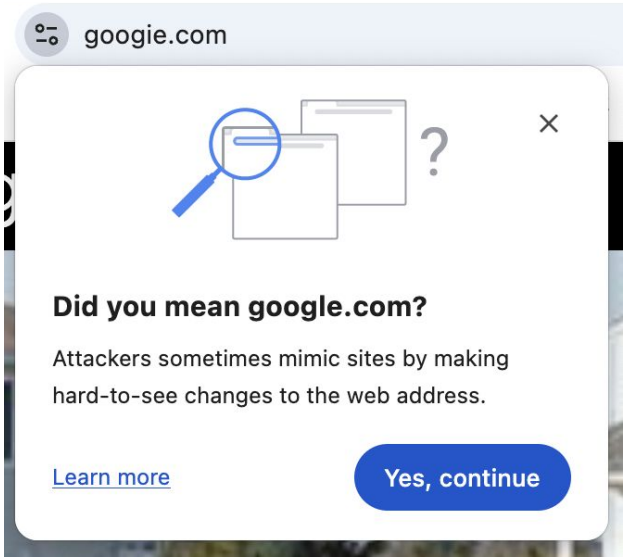


Client ID Reputation

An AS can keep track of “domain health”, handling the consent steps differently based on factors such as:

- How many client ID URLs are used per domain (is it a spammy site)
- Has more than one user authorized the domain
- Re-prompt for consent if the WHOIS data has changed
- Does the domain “look” similar to other popular domains?

There are many tools for domain reputation today!



Handling first-seen use of Client ID URL

- Incrementally lowering the guard on the consent screens as described previously
- Associate client with the user authorizing, instead of trying to tie it to a developer identity
- Apply greater rate limits to access tokens issued to this client

Planned Updates

Based on feedback from implementers and during call for adoption, discussed so far on GitHub

- Additional guidance of handling new client IDs as just described above
- Additional guidance for SSRF mitigation
- Restrict URIs in metadata (logo, etc) to `https` URLs (except redirect URI)
- Additional security considerations for changes in client metadata
- Prohibit following redirects when fetching metadata document

SSRF Mitigations

Issue #30

- Only fetch metadata after authenticating the user
  (prevents fetches on unauthenticated requests to the authorization endpoint)
- Send an `Accept: application/json` header when fetching (PR #51)
  (enables early rejecting of requests to images, HTML pages, etc)
- Avoid fetching if DNS name resolves to private use IP addresses (PR #49)

Restrict URI properties to absolute `https` URLs

PR #50

Properties like `client_uri`, `logo_uri`, etc MUST be `https` URLs

Avoids potential issues with URIs like `javascript:` or `shortcuts:` if the AS evrer renders links or embeds the URLs

Security considerations for changes in metadata

PR #47

“Significant changes to client metadata may affect the trust relationship between the authorization server and the client, and could impact the validity of previously granted user consent.

Authorization servers may choose to invalidate existing grants, require fresh user consent, or implement other policies when certain types of metadata changes are detected.”

Prohibit following redirects when fetching metadata

PR #46

Following redirects would result in a `client_id` value in the metadata different from the URL you fetched, which is invalid.

Further Discussion

https://github.com/oauth-wg/draft-ietf-oauth-client-id-metadata-document/issues