IETF 121
Dublin
November 2024

Aaron Parecki
Emelia Smith

Client ID Metadata Document

https://datatracker.ietf.org/doc/draft-parecki-oauth-client-id-metadata-document/
draft-01

OAuth Client Registration

OAuth clients need to register with the Authorization Server to establish things like:

- Redirect URI
- Name
- Logo
- Scopes
- Client Authentication Methods
- etc

OAuth Client Registration

Pre-registration 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, where there is no single central server to register the OAuth client

Dynamic Client Registration

Dynamic Client Registration (RFC7591) is an option, but provides additional operational challenges:

- The AS has to maintain all received registrations for an indeterminate amount of time
- This has led to some AS's building a "cleanup" process that removes inactive clients
- Clients have no way to know if they've been "cleaned up"
  - RFC 6749 says the AS MUST NOT redirect on invalid client ID error
  - There is no standard "check client credentials" method, nor would this scale well
  - Not all clients have client credentials (i.e., public clients without a client_secret)
- This leads to a dead end for the user, leaving users confused and stranded
- Which then leads to clients doing Dynamic Client Registration on every user authorization to avoid the dead end situation
  - Which creates an application management problem for users and administrators, reducing security,
  - And uses an excessive amount of storage as the same client is stored multiple times.
    (one real example has an instance with ~40,000 users storing ~400,000 clients)

See this thread for an example of this discussion https://github.com/mastodon/mastodon/issues/27740

Previously Proposed Solutions

draft-looker-oauth-client-discovery-01 (Issue #5)

- https://www.ietf.org/archive/id/draft-looker-oauth-client-discovery-01.html
- Uses a .well-known endpoint of `/.well-known/oauth-client`
- Has the well-known problem of multi-tenant systems, e.g.

  `/.well-known/oauth-client/client1` VS
  `/client1/.well-known/oauth-client` VS
  `/tenant1/.well-known/oauth-client/client1` VS
  `/.well-known/oauth-client/tenant1/client1`

- Doesn't allow for the document to be hosted in a location that differs from the `client_uri`, preventing usage by e.g., mobile apps.
- Doesn't allow for non-public apps, e.g., apps on a corporate intranet that connect to a remote AS
  (though this isn't the general use-case for client_id as URIs)

Client ID Metadata Document

- The client publishes their metadata (Dynamic Client Registration vocabulary) at a URL
  - https://www.iana.org/assignments/oauth-parameters/oauth-parameters.xhtml#client-metadata
  - Doesn't have to be at an 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 OAuth flow
  - `…&client_id=https://example-app.com/client.json&scope=…`

Client ID Metadata Document

These MUST match

(like in Protected Resource Metadata)

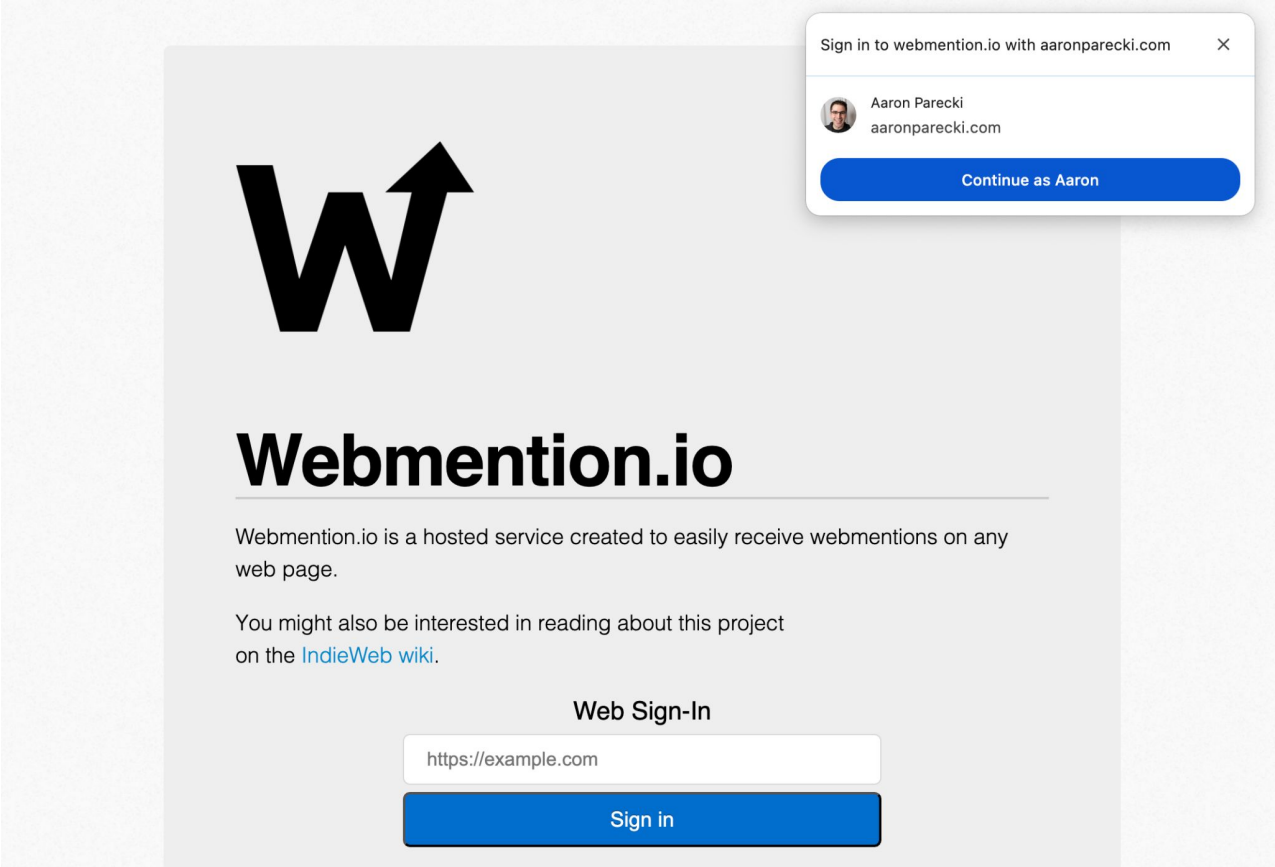
```
…&client_id=https://webmention.io/id&...

{
  "client_id": "https://webmention.io/id",
  "client_name": "Webmention.io",
  "client_uri": "https://webmention.io",
  "logo_uri": "https://webmention.io/img/webmention-logo-380.png",
  "redirect_uris": [
    "https://webmention.io/auth/callback"
  ]
}
```

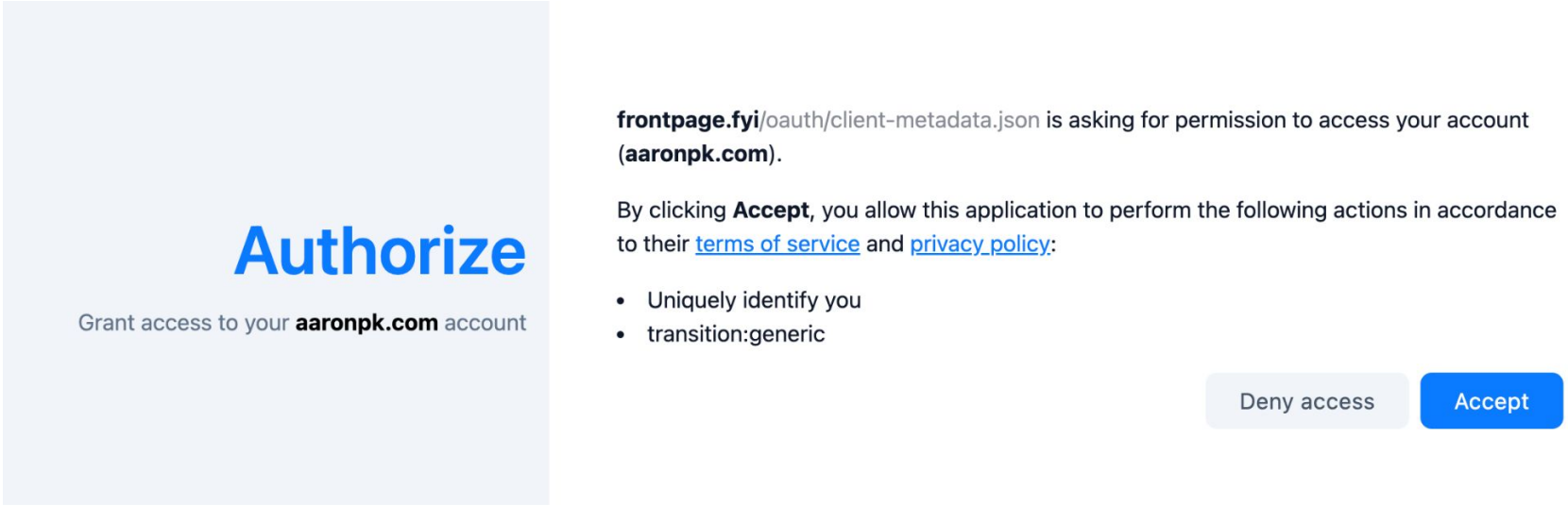
Client ID Metadata Document

- The AS can fetch the metadata to display the client name and logo
  - Or it can choose to ignore it
  - AS should validate data within the document according to their security practices that would be applicable for Dynamic Client Registration
- Client metadata includes `redirect_uris`
- Client metadata can include a public key so the client can authenticate requests to the token endpoint and elsewhere
- Client metadata documents MUST be publicly accessible, such that the AS can fetch them.

Implementation with FedCM



New Implementations: BlueSky and apps!



https://docs.bsky.app/blog/oauth-atproto

frontpage.fyi
smokesignal.events

Implementations

- Based on an early version of IndieAuth, now referenced by IndieAuth
  - indieauth.spec.indieweb.org
- Based on Solid-OIDC Client ID Documents
  - Solid-OIDC defines additional JSON-LD requirements, but should otherwise be compatible
  - Implemented in Enterprise Solid Server from Inrupt (via prior art)
- Live Implementations:
  - Clients: **smokesignal.events**, **frontpage.fyi**, webmention.io, indiebookclub.biz, indielogin.com
  - AS: Micro.blog, WordPress IndieAuth Plugin, ProcessWire IndieAuth Plugin, aaronparecki.com
  - **New AS: BlueSky** https://docs.bsky.app/blog/oauth-atproto
- Planned:
  - AS: Mastodon https://github.com/mastodon/mastodon/issues/31151
  - Clients: phanpy.social and other 3rd party Mastodon apps.
- Used in the OAuth profile for FedCM when used by decentralised services
  - https://github.com/aaronpk/oauth-fedcm-profile

What’s Next

- Recommendations for localhost/development clients
- WG Adoption?

Open Questions

- Require that `client_uri` is a prefix of `client_id`? (Issue #10)
- Recommendations when an AS sees a client has changed its keys (Issue #11)
  - e.g. revoke consents
- Best way to handle localhost/development clients (Issue #12)
- Require all URIs to be absolute and prohibit data: URIs? (Issue #18, #19)
- How can caching happen? How does invalidation happen? (Issue #3)