OIDCFed status update

Davide Vaghetti
GÉANT OIDCFed Team (GN4-2 JRA3 Task 3 1.A)

Consortium GARR

eduGAIN SG
Mar 27th, 2018
OIDCFed Team Activities

• Development of OIDC Federated Client
  • Python library
  • Android and IOS POC
  • PHP POC
• Development of OIDC Federated Provider
  • Python library
  • SaToSa Frontend
  • Shibboleth OIDC Extension
• Development of OIDC Federation tools
  • Metadata Signing Service
• Development of OIDC Federation profiles
  • OIDC Federation draft implementation profiles
  • OpenID Foundation OIDC for Research and Education working group (currently setting it up)
• OIDC Federation pilot

Please check out

• [https://wiki.geant.org/display/gn42jra3/T3.1A+OpenID+Connect+Federation](https://wiki.geant.org/display/gn42jra3/T3.1A+OpenID+Connect+Federation)
• mailing-list: [oidcfed@lists.geant.org](mailto:oidcfed@lists.geant.org)
OIDCFed Team

- Maarten Kremers - Task leader
- Roland Hedberg - Principal developer and OIDC Federation standard editor
- Davide Vaghetti - Sub task leader
- Ioannis Kakavas - previous sub task leader (left)
- Alejandro Perez Mendez (left)
- Peter Schober
- Janusz Ulanowski
- Janne Lauros
- Henri Mikkonen
- Juha Hopia
- Andreas Åkre Solberg
- Elena Torroglosa
- Constantin Scifos
- Alexandru Cacean
- Hervé Bourgault
OIDC Federation: the problem space
OIDC: Actors

• The **User** who wants to access a protected resource, either by himself or through an application.
• The **Relying Party** (often called the Client) is the entity that will request and use an access token.
• The **OIDC Provider (OP)** is the entity that will release the access token.
OIDC: OP and RP needs to know about each other

RP redirect the user to the OP’s authorization_endpoint

OP redirect the user to the RP’s redirect_uri

RP exchange the user code for an access_token at the token_endpoint (and authenticate…)

IIRP requests user claims at the OP’s userinfo_endpoint
The RP receives and consumes the OP metadata (provider configuration) that are self-asserted.

No trust anchor is provided.
The OP receives a client registration request from the RP. The information provided by the RP is self-asserted.

No trust anchor is provided.

The OP sends a client registration response to the RP, once again all the information is self-asserted.

No trust anchor is provided.
This document describes **how an identity federation can be built around a trusted third party, the federation operator.**

**Metadata:**

- **signing_keys:** A JSON Web Key Set (JWKS) representing the public part of the entity's signing keys.
- **metadata_statements:** JSON object where the names are federation identifiers and the values a signed JSON documents containing compounded metadata statements rooted in that federation. There is one value per name.

[Link to spec](http://openid.net/specs/openid-connect-federation-1_0.html)
Deploying multiple R&E communities with OIDCFed

https://github.com/OpenIDC/fedoidc/blob/master/doc/howto/multifederation.md

Outcome of the two day OIDCFed design meeting in Amsterdam in January 2018 (cudos to Alejandro Pérez Méndez)

Key elements:

- A metadata signing service for each federation
- Communities of federations can stand for:
  - Interfederation services (aka eduGAIN)
  - Entity categories
The SWAMID profile for a OpenID Connect federation


A recent elaboration of Roland Hedberg and the SWAMID Federation Operators.

Key elements:

- Direct relationships between the Federation and the final entities (RPs and OPs)
- No metadata_statements passed by value, only metadata_statements_uri
- All metadata_statements_uri used for registration and configuration providing are served by the Metadata Signing Service operated by the Federation

Comments and feedbacks are welcome!

(open a github issue, or make a PR)