Keycloak & OpenID Federation: Empowering Dynamic Trust in Federated Environments

28 August 2025 Keyconf 25
Amsterdam Netherlands

RCIAM
Identity Access Management for Research Communities

grnet

RCIAM

Identity Access Management
for Research Communities

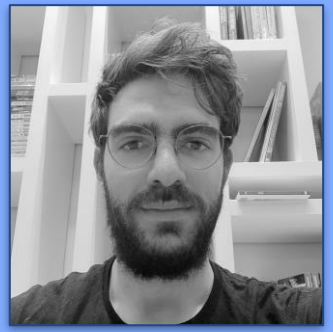
- Team in Directorate of European and International Infrastructure Projects of **GRNET**
- Providing facilitating seamless access to research resources for diverse stakeholders within the European Open Science Cloud (EOSC) based on Keycloak
- Researchers, academics, policy makers, funders, innovators, citizens and public actors could securely access Open Science services across infrastructures

CORE TEAM

- Konstantinos Georgilakis
DEVELOPER (SPEAKER)

- Nicolas Liampotis
SERVICE MANAGER

- Nick Mastoris
DEVELOPER



- Andreas Kozadinos
DEVELOPER

- Halil Adem
DEV OPS

OpenID Federation

Empowering Dynamic Trust in Federated Environments

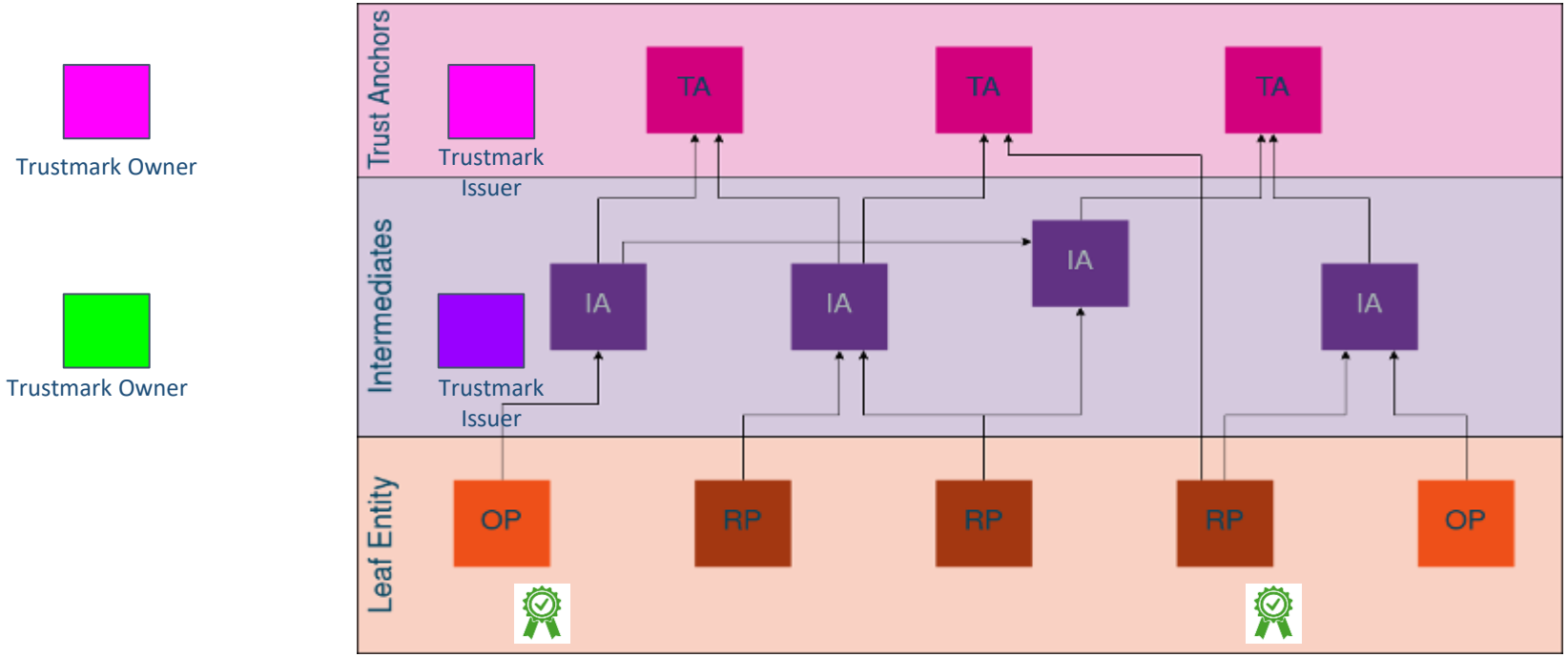
OpenID Federation: Empowering Dynamic Trust

- ★ A robust way for establishing dynamic trust between OpenID Providers (OPs) and Relying Parties (RPs)
- ★ Based on OAuth2/ OIDC protocols
- ★ Simplifies management of large-scale identity federations
- ★ Eliminates cumbersome manual or bilateral trust agreements
- ★ Enables secure interactions authenticated via Trust Anchors

OAuth2/OIDC vs OpenID Federation

Feature	Classic OAuth2/OIDC	OpenID Federation
Trust Establishment	Manual / Bilateral	Dynamic / Hierarchical
Metadata Exchange	Static JSON URLs	Signed Entity Statements
Trust Lifecycle	Admin-managed	Self-validating & verifiable
Scalability	Limited	Internet-scale
Security Assurance	Varies	Enforced via trust marks

OpenID Federation - Entity Roles



Source: 'Trust & Identity Incubator' presentation in GEANT T&I Incubator Public Sprint demo #3.1

Key Concepts of OpenID Federation

- ★ Entity Configuration: Each participant self-describes metadata in a signed JWT
- ★ Entity Statement: Trust anchors or authorities issue signed assertions about entities
- ★ Trust Chain: Sequence of signed JWTs linking an RP/OP to a trust anchor

Explicit Registration

- Pre-approved metadata exchange
- No changes in OIDC/ OAuth2 flows
- Federation with specific metadata requirements

Automatic Registration

- RP dynamically registers with OP via federation metadata
- Dynamic trust resolution
- Large federation with many RPs

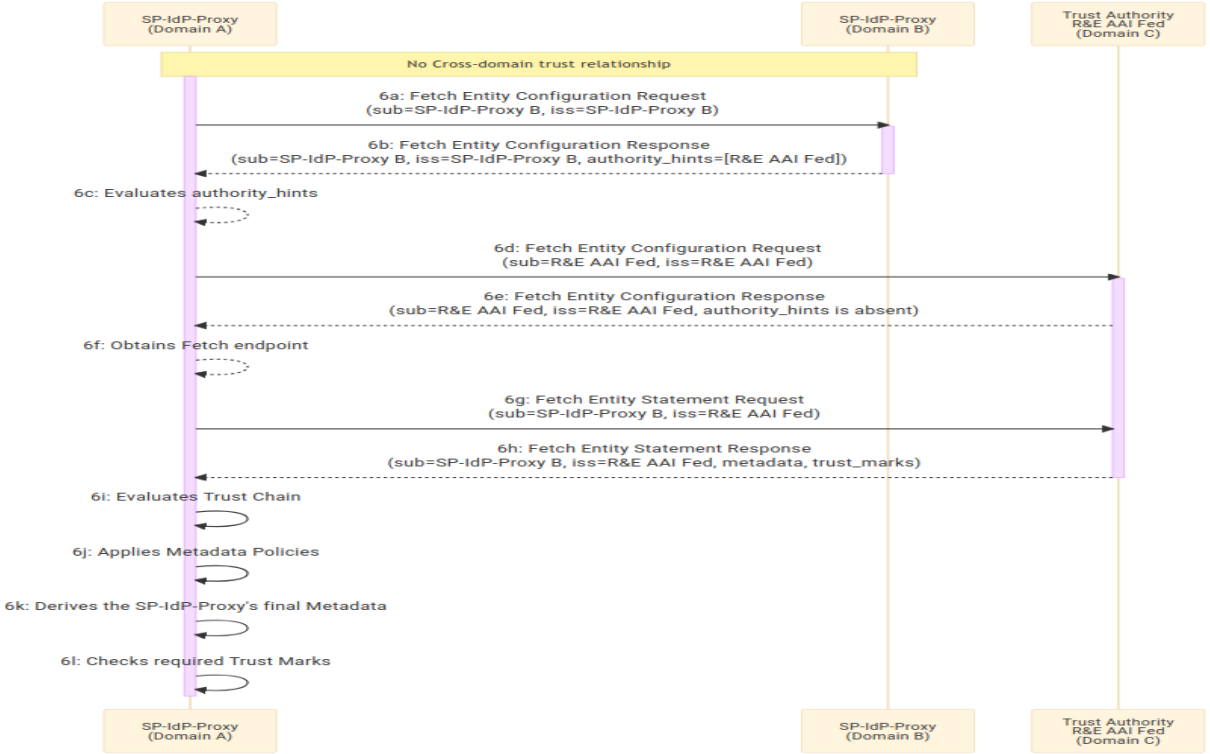
Explicit vs Automatic Registration

	Explicit	Automatic
Pros	● More control ● Policy checks and pre-registration processing	● No need for prior contact ● Fully dynamic
Cons	● Client-side registration logic requirement ● Periodical re-registration need	● Less initial control ● Must be able to validate on the fly

Trust Chain Resolution & Validation

- All metadata is delivered as signed JWTs
- Trust chain built by recursive resolution of authority_hints up to the trust anchor
- Metadata is valid only if:
  - JWT signatures are valid
  - All trust chain policies are satisfied
  - Authority hint fetch endpoint consists subordinate entity
  - Leads to an acceptable common trust anchor
  - Valid Trust Marks

Trust Chain Resolution & Validation (2)



Source:https://aarc-community.org/guidelines/aarc-g100

Federation Policies

- Rules and constraints for Entities within a federation
- Ensure trust and interoperability
- metadata_policy claim of fetch endpoint
- Cascade to all Subordinate Entities, ensuring consistent metadata across the federation
- Applied in Trust Chain Resolution

Trust Marks – Assurance & Policy Binding

- Signed evidence of conformance to specific requirements or certifications
- Signed JWTs issued by Trust Mark Issuers
- Prove conformance with:
  - Security requirements
  - Data privacy policies
  - Interoperability profiles (e.g. REFEDS)
- Automate compliance validation across federated infrastructure

OpenId Federation in Keycloak

Keycloak Epic #40509

- Support Keycloak being RP and OP
- Explicit Registration
- Automatic Registration
- Federation Policies
- Trust marks
- Endpoints Cache

OpenID Federation Configuration

General Settings

OpenID Federation Enabled — On

Authority Hints * — https://ta.example.org

Add Authority Hint

Federation Lifespan — 1 Days

Contacts — check-in@egi.eu

Add Contact

Logo URI — https://ta.example.org/icon.jpg

Policy URI — https://ta.example.org/privacy/en

Organization Name — EGI Foundation

Trust anchor configuration

Trust Anchor Settings

Add Trust Anchor

1 - 1

Trust Anchor	Entity Types	Client R...
https://ta.example.org	OPENID_PROVIDER	EXPLICIT

Trust Anchor Configuration

- Trust Anchor
- OP - RP
- Automatic - Explicit Registration
- RP default configuration
- Trust Marks

.well-known/openid-federation JWT endpoint


```
{
  "exp": 30157523432,
  "iat": 30157660232,
  "jwks": { "keys": [ { ... }]  },
  "metadata": {
    "openid_provider": {
      "contacts": [ "check-in@rciam.example.org"]
      "organization_name": "RCIAM Foundation",
      "client_registration_types_supported": ["explicit"],
     "registration_endpoint": "https://rciam.example.org/auth/realms/rciam/openid-federation/clients-registrations"
       ...
    },
    "openid_relying_party": {
     "grant_types": ["authorization_code"],
     "response_types": ["code"],
     "application_type": "web",
      "redirect_uris": ["https://rciam.example.org/auth/realms/rciam/broker/federation-endpoint"],
      "client_registration_types": ["explicit"],
     "subject_types_supported": ["public","pairwise"]
    }
    "federation_entity": { ...}
  },
  "iss": "https://rciam.example.org/auth/realms/rciam",
  "sub": "https://rciam.example.org/auth/realms/rciam",
  "typ": "entity-statement+jwt",
  "authority_hints": ["https://trust-anchor.sandbox.eosc.grnet.gr"]
}
```


Signing keys

Same as .well-known/openid-configuration

Common Redirect Uri

Trust Infrastructure

- Deploy Trust Anchor / Intermediate based on lighthouse/golang library
- Deploy Test RP based on OFFA, resolving OPs from the Trust Anchor. Suitable for testing automatic registration.
- Pilot in the context of EC-funded project EOSC Beyond for enabling scalable, policy-driven trust establishment and dynamic discovery of trusted entities based on AARC-G100 guidelines

eosc | BEYOND

Trust Anchor Rest API

- /list : entities enrolled in TA
- /fetch : information about an entity
- /resolve : metadata of entity with trust chain
- /enroll /enroll_requests : Automatic registration based on checkers
- /trustmarks : available Trust Marks operation

OpenID Federation Identity Provider (OP)

OpenID Federation Settings

Authority Hints

https://trust-anchor.sandbox.eosc.grnet.gr

Trust Anchor ID

https://trust-anchor.sandbox.eosc.grnet.gr

Expiration Time

8/29/2025, 12:07:33 PM

- Similar to OIDC Identity Provider
- Created with explicit registration based on RP issuer and trust anchor ID
- These values as long as Client id, Client Secret retrieved from OP explicit registration response
- Disable in expiration time
- Common Redirect uri

OpenID Federation Client (RP)

General Settings

Client ID *

https://rciam.example.org/test

Name

EGI Foundation

Description

Expiration Time

8/29/2025, 12:07:33 PM

- OIDC Client
- Created with federation registration endpoint
- Disable in expiration time
- Being able to be updated with federation registration endpoint
- OpenID Federation access policies

Future Tasks

- Finalize explicit registration
- Automatic Registration OP
  - Authorization Code
  - Token Introspection
- Automatic Registration RP
- Federation Policies (experimental feature now)
- Trust marks

PR for OpenID Federation explicit OP

OpenId Federation OP back end #41419

Edit | <> Code

Draft | cgeorgilakis wants to merge 1 commit into `keycloak:main` from `eosc-kc:40511_openid_federation_op`

Conversation 12 | Commits 1 | Checks 80 | Files changed 85 | +3,630 −363

cgeorgilakis commented last week — Contributor

Closes #40511

This is the initial PR for supporting OpenID Federation with Keycloak being an OP and explicit registration, accepting only Entity Statement. Ui will be a separate PR, documentation will be enhanced based on it. Some code parts are taken from the OpenID Federation extensions, taking into account that we are in core Keycloak, needed bug fixes and code improvements , compliant with latest OpenID Federation specification version.

Some key points for the PR:

- OpenID Federation is disabled by default and can be enabled with required configuration (more in ui part). Inspiration : https://www.authlete.com/developers/oidcfed/#configuring-federation
- Keycloak as a proxy can participate in more than one Federation(Trust Anchor) with different configuration options especially for RP part, trust marks etc (future features). That's why it is separated table.
- Tests for explicit client registration has been done with a real test OpenID Federation(Trust Anchor, RP) via the help of golang lighthouse OpenID Federation implementation
- I have added tests for Federation Entity Configuration. For adding testing for explicit registration we need to mock REST API for Trust Anchor Entity Configuration, fetch and list endpoints. Take into account that they are JWT signed statement. Moreover, we need to create a mock jwt valid during test execution for client explicit registrations. How could we develop it? I have only a mock Java code for create a RP entity Statement based on mock keys. Should it be a separate issue?

For more opened issues related to explicit client registration in an OP, you could see the epic issues.

cgeorgilakis requested review from **a team** as code owners last week

Reviewers
pruivo
Requested changes must be addressed to merge this pull request.

Assignees
No one assigned

Labels
team/cloud-native team/core-clients team/core-iam team/sre

Projects
None yet

Milestone
No milestone

Development
Successfully merging this pull request may close these issues.
OpenID Federation OP with explicit registration

Thanks!

Does anyone have any questions?

faai@grnet.gr

RCIAM
Identity Access Management
for Research Communities

grnet