OAuth2 Flows and Wallets

Nuts supports 2 OAuth2 flows for acquiring an access token. The service-to-service flow and the user flow.

Service-to-Service flow

The service-to-service flows is for data exchanges that don't require the presence of a (human) user. Credentials that are presented during this flow are subject to legal organizations (e.g. registered care organizations).

This flow uses a custom grant type called `vp_token-bearer`. Presentation requests always and only target `organization` wallets.

The flow is secured with DPoP (optional). See "Security controls" for a detailed description.

When to use

Data exchanges for which this flow is suitable are background processes or exchanges that aren't subject to GDPR (or other local privacy regulations).

User flow

The user flow us for data exchanges that require the presence of a (human) user. Credentials that are presented during this flow are typically a combination of:

- credentials subject to a legal organization, and
- credentials subject to a natural person, registered caregiver or employee of a legal organization.

At the time of writing, there is no governing body that issues Verifiable Credentials to (human) users, meaning the only viable option is an employee-type credential that is a self-attestation (by the organization) that the current user is an employee.

A typical example of requested credentials are a legal care organization and self-attested `EmployeeCredential` that is issued by the care organization.

This flow uses OpenID for Verifiable Presentations (OpenID4VP), draft 21 at the time of writing.

The flow is secured with JAR, PKCE and DPoP (optional). See "Security controls" for a detailed description.

When to use

Data exchanges for which this flow is suitable are ones for which the data holder/receiver requires an identity of the end-user for logging means (Dutch norm NEN-7510) and/or GDPR compliance.

Security controls

The following security controls are used by the OAuth2 flows:

- JWT-Secured Authorization Request (JAR, RFC9101) provides integrity protection and authenticity for the credential presentation request. Usage is enforced by the server.
- Proof Key for Code Exchange (PKCE, RFC7636) provides authenticity of the client retrieving the access token. This mitigates a MITM stealing authorization codes. Usage is enforced by the server.
- Demonstrating Proof of Possession (DPoP, RFC9449) provides authenticity of the client using the access token. This mitigates a MITM stealing access tokens. Usage is optional, to be enabled by the client.

Revision #7
Created 25 April 2024 12:22:15 by Rein Krul
Updated 25 April 2024 14:07:01 by Rein Krul