OAuth2 Scopes and Presentation Definition Mapping

Scope design

When designing a system that uses OAuth2, you have to decide how scopes map to resources that the client will attempt to access. "Resource access" is typically a specific REST-style HTTP operation on a specific URL, e.g. `POST /products/staplers/1`. Things to consider when designing scopes are discussed in this section.

Broad v.s. narrow scopes

Broad scopes are generally high level e.g., a scope that gives access to a certain use case or larger group of resources. Narrow scopes are often low-level e.g., a scope that gives read access to a specific resource, limited set of resources or operations. Examples scopes for an employee that's authorized to buy supplies for their employer:

- Very broad: `buyer`
- Broad: `buyer office` (office supplies only)
- Broad: `buyer lt-1000` (orders less than 1000 euros)
- Narrower: `buyer office:staplers` (staplers only)
- Very narrow: `buyer office:staplers:red lt-10` (red staplers only, less than 10 euros)

How scopes are mapped to operations on resources influences:

- How often clients need to request a new access token, if the previous token does not give access to a required resource.
- When access to a specific resource is authorized: when the access token is issued, or when it's used.

Broad, high-level Scopes

High-level, broad scopes typically give access to an entire use case, service, or group of resources. Checks that are executed before an access token is issued are limited to the Verifiable Credentials the client can present.

- Identification and authentication (user/client identity)
- General user access to the functionality (e.g. is admin, can buy supplies, etc.)

A real-life example of a broad scope is the Nuts eOverdracht use case, which specifies the following scopes:

- `eOverdracht-sender` which gives access to the receiver's services required by a care organization that wants to transfer a patient to another organization.
- `eOverdracht-receiver` which gives access to the sender's services to the transfer receiver.

However, when a resource is accessed, the system needs to verify that the scope gives access to the specific resource operation.

This type of scope is supported by the Nuts node.

Narrow, low-level Scopes

Narrow, low-level scopes typically give access to specific operations on specific resources, e.g., reading a specific patient's medical summary.

> Narrow scopes with an AuthZen policy decision point are supported from Nuts node v6.3.

The Nuts node supports narrow scopes next to a broad scope. The vendor runs an AuthZen policy decision point (PDP) that decides on them:

- The client requests the broad scope together with the narrow scopes, e.g. `eOverdracht-sender patient:123:read`. The broad scope is the one with a credential profile in the policy file.
- In the policy file, the broad scope sets `scope_policy` to `dynamic`.
- The Nuts node checks the presented credentials against the broad scope's presentation definition. It then asks the PDP (`policy.authzen.endpoint`) about every requested scope, and sends it the values from the credentials: the fields with an `id`, the same ones as in the token introspection response.
- If the PDP denies the broad scope, the request is denied. The other scopes are granted only when the PDP allows them.

Things to consider:

- This leads to more access tokens, since each token has a more limited use. If user authentication involves manual input (e.g. presenting a credential with a mobile wallet), the user experience gets worse.
- Authorization decisions move to the moment the token is issued, and the vendor has to run and maintain the PDP.

Without a PDP, an access token request can't contain other scopes (`profile-only`, the default), or the other scopes are granted without checks (`passthrough`).

Policies: Mapping Scope to Authentication Subject

Due to the ongoing development of personal authentication methods and associated protocols, the Nuts node currently only supports the OAuth2 `vp_token` grant for production. User authentication via OpenID4VP is experimental and not intended for production. It's still required to pass user

claims within the token request if a data exchange contains PII (Personally Identifiable Information) and/or medical data (e.g., Social Security Number or EHR records)

Mapping document

This section contains an example of a presentation definition mapping document as it could be specified by a use case. The Presentation Definition is described more in detail in AuthN using Verifiable Credentials.

```
{
  "zorgtoepassing": {
    "organization": {
      "format": {
        "ldp_vc": {
          "proof_type": [
            "JsonWebSignature2020"
          ]
        },
        "ldp_vp": {
          "proof_type": [
            "JsonWebSignature2020"
          ]
        },
        "jwt_vc": {
          "alg": [
            "ES256"
          ]
        },
        "jwt_vp": {
          "alg": [
            "ES256"
          ]
        }
      },
      "id": "pd_any_care_organization",
      "name": "Care organization",
      "purpose": "Finding a care organization for authorizing access to medical metadata",
      "input_descriptors": [
        {
          "id": "id_nuts_care_organization_cred",
```

```
          "constraints": {
            "fields": [
              {
                "path": [
                  "$.type"
                ],
                "filter": {
                  "type": "string",
                  "const": "NutsOrganizationCredential"
                }
              },
              {
                "id": "organization_name",
                "path": [
                  "$.credentialSubject.organization.name",
                  "$.credentialSubject[0].organization.name"
                ],
                "filter": {
                  "type": "string"
                }
              },
              {
                "id": "organization_city",
                "path": [
                  "$.credentialSubject.organization.city",
                  "$.credentialSubject[0].organization.city"
                ],
                "filter": {
                  "type": "string"
                }
              }
            ]
          }
        }
      ]
    }
  }
}
```

What the use case decides

The use case's policy file sets, per scope:

- which credentials the care organization must present (`organization`);
- which credential fields the data holder gets in the token introspection response: each field with an `id` becomes a claim. Give every field the data holder uses for authorization or logging its own `id`;
- optionally, which credentials the service provider must present (`service_provider`, in development).

How the Nuts node reads the policy file is described in Access Token Policy in the Nuts node documentation. That covers the file format, introspection claims, returning part of a value, the two-VP flow and `credential_selection`.

Revision #13
Created 2024-04-24 07:08:03 UTC by Rein Krul
Updated 2026-11-06 16:20:08 UTC by Rein Krul