Information und Technik
Nordrhein-Westfalen

Abgesicherte OGC API Dienste

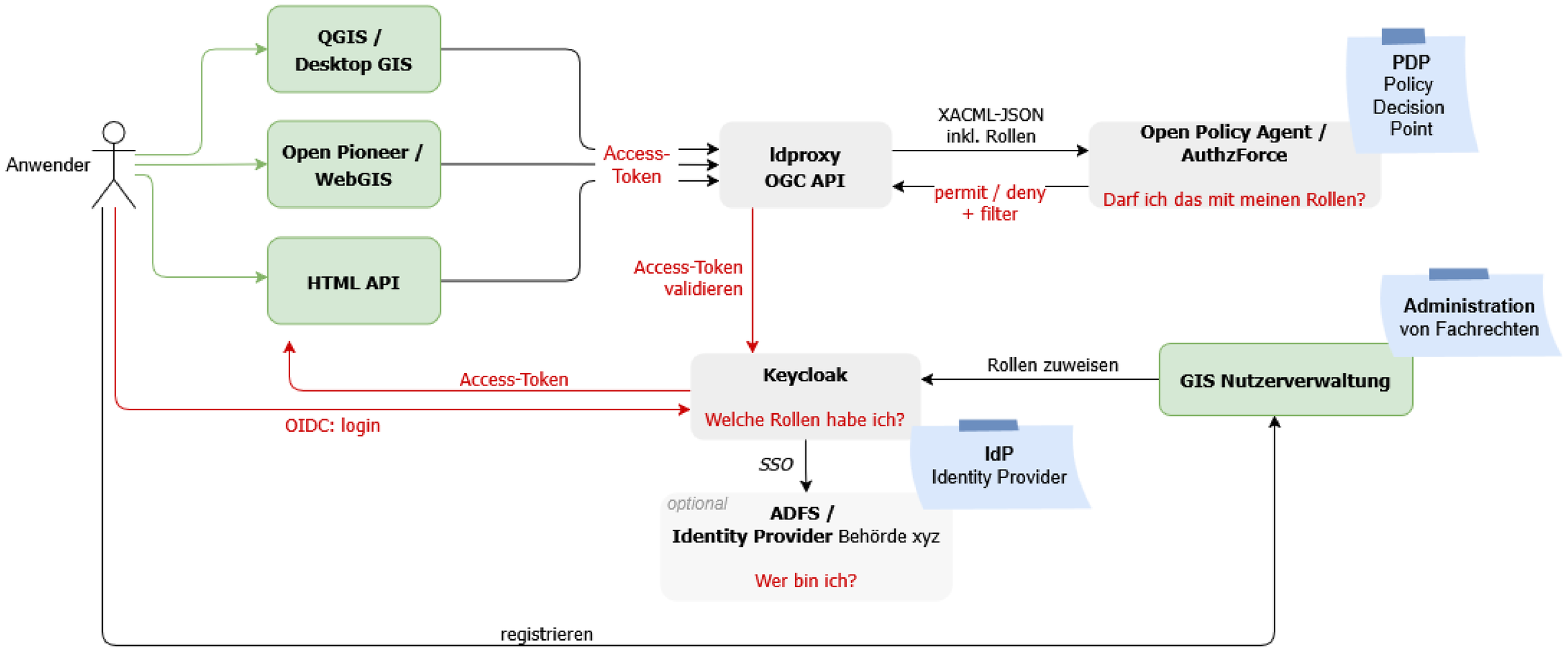
Praktische Umsetzung mit Keycloak bei IT.NRW

17. März 2026

Agenda

1. Übersicht der beteiligten Komponenten
2. Grundregeln für Zugriffstoken
3. Konfiguration von Keycloak und ldproxy
4. Rollenbasierte Rechtesets im Policy Decision Point (PDP)
5. OAuth2/OIDC Konfiguration in QGIS
6. Fazit

Infrastruktur



Zugriffstoken

Berechtigungen begrenzen

- **Grundregel:** Zugriffsrechte auf das notwendige Minimum beschränken.
- **Zielgruppe (Audience / aud):** Tokens explizit an definierte Ressourcenserver binden (Wo darf das Token genutzt werden?).
- **Berechtigungen (Scope / scope):** Zugriff auf spezifische Ressourcen und Aktionen (z. B. read vs. write) begrenzen.

Fazit: Die Sicherheit basiert auf der strikten Validierungspflicht durch den Ressourcenserver.

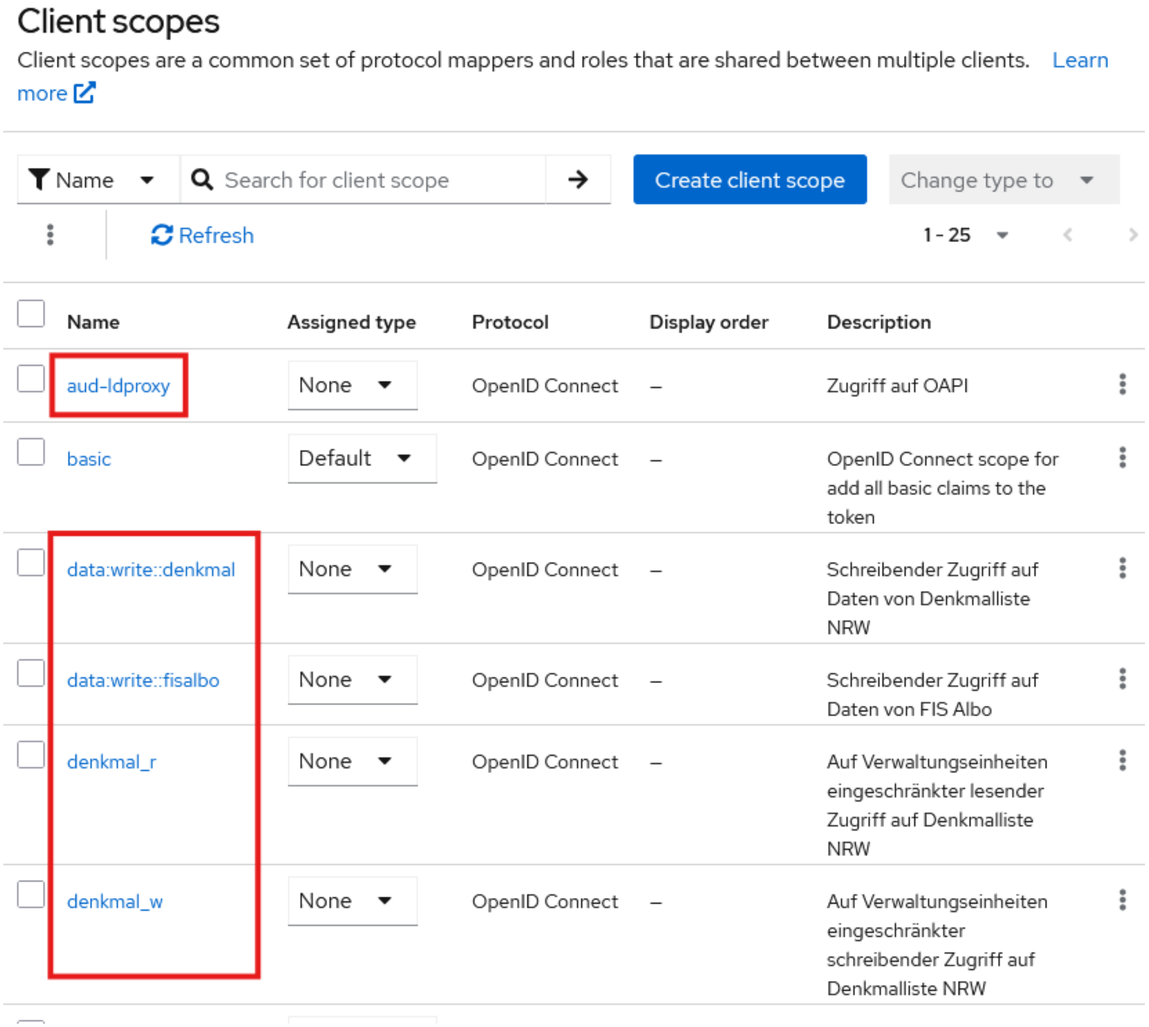
```
"typ": "Bearer",
"resource_access": {
  "denkmal": {
    "roles": [
      "ratingen_r",
      "ratingen_w",
      "duesseldorf_r",
      "duesseldorf_w",
      "read::denkmal",
      "data:write::denkmal"
    ]
  },
  "fisalbo": {
    "roles": [
      "duesseldorf_w",
      "duesseldorf_r",
      "read::fisalbo",
      "data:write::fisalbo"
    ]
  }
},
"aud": [ "https://lv.gistest.nrw.de/ldproxy" ],
"scope": "openid email profile read", // > read::denkmal
"email_verified": true,
```


https://datatracker.ietf.org/doc/html/rfc9700#section-2.3

Keycloak

Client Scopes

- Client Scopes sind gruppierte Sammlungen von Protokoll- und Rollen-Mappings und dienen als wiederverwendbare Vorlage für die Konfiguration von Token-Inhalten.
- Ermöglicht eine einheitliche Konfiguration über die gesamte Infrastruktur.
- Jeder Client erhält präzise nur die Daten, die er für seinen Anwendungsfall benötigt.
- Änderungen am Scope wirken sich sofort auf alle verknüpften Clients aus.

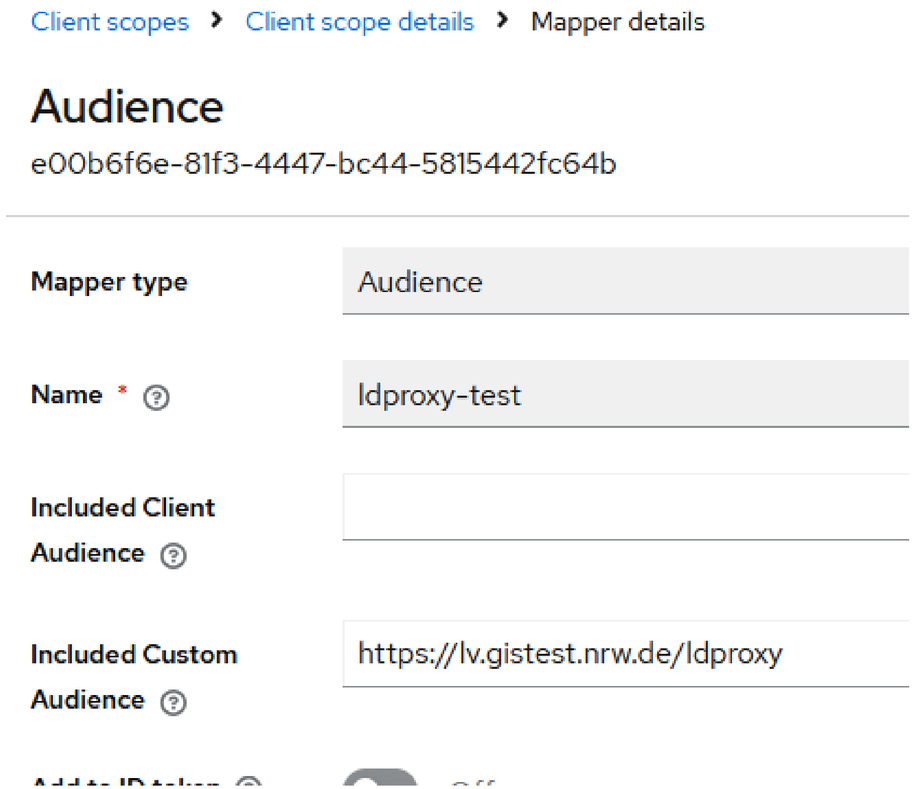


https://www.keycloak.org/docs/latest/server_admin/index.html#_client_scopes

Keycloak

Audience Mapper

- Der Audience-Mapper legt fest, für welche Empfänger (Ressourcenserver) ein Token gültig ist.
- **Statischer Wert:** Eine fest definierte URL oder Kennung
- **Client ID:** Automatische Übernahme der Client-ID als Audience



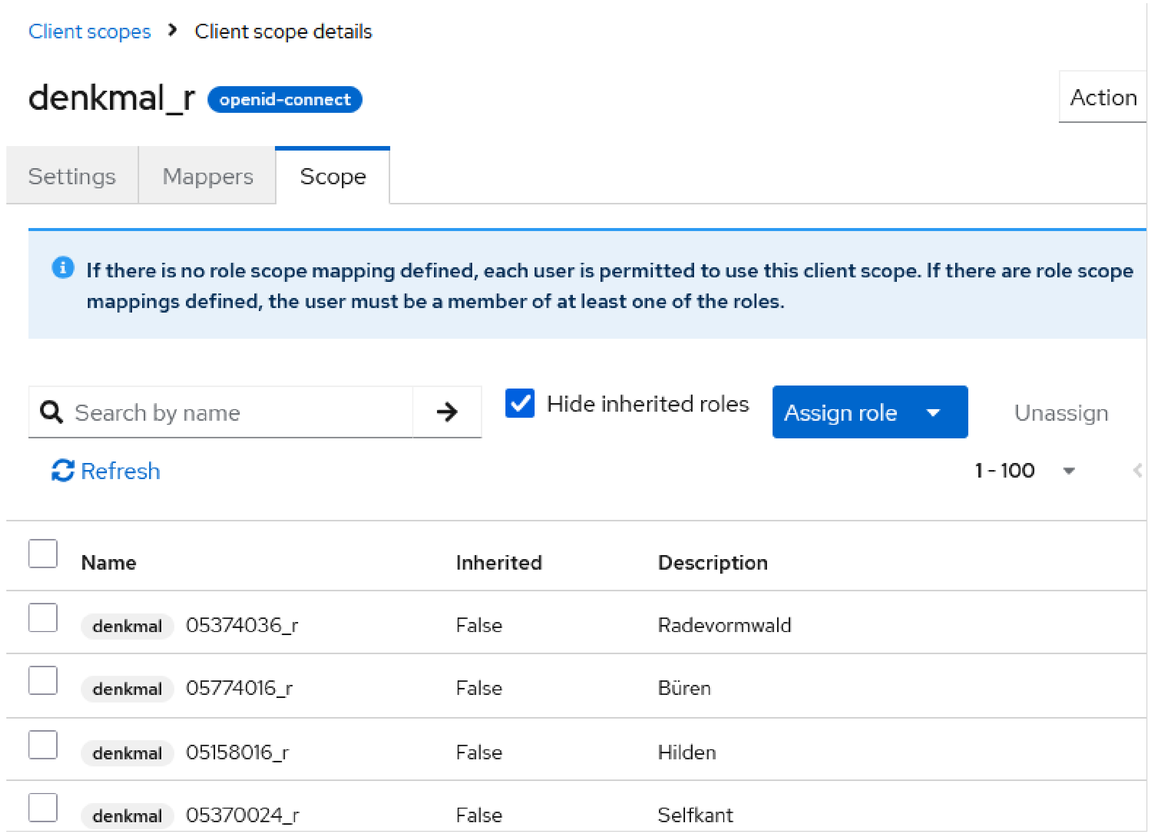
https://www.keycloak.org/docs/latest/server_admin/index.html#_audience_hardcoded

Keycloak

Role Scope Mapping

- Role Scope Mappings begrenzen die im Token enthaltenen Rollen.
- Das ausgestellte Token enthält nur die Rollen, die explizit für den Scope des Clients definiert wurden.
- Die Berechtigungen des Tokens können damit gezielt eingeschränkt werden.

Achtung: Um eine effektive Rollen-Limitierung zu erreichen, muss **Full Scope Allowed** deaktiviert werden.

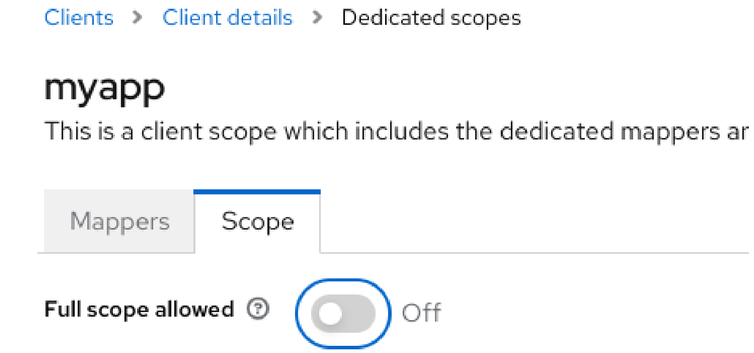


https://www.keycloak.org/docs/latest/server_admin/index.html#_role_scope_mappings

Keycloak

Full Scope Allowed

- Eine Konfiguration auf Client-Ebene, die steuert, welche Rollen eines Benutzers in dessen Token aufgenommen werden.
- *ON (Standardeinstellung):* Das Token enthält alle verfügbaren Realm- und Client-Rollen des Benutzers ungeachtet der zugewiesenen Scopes.
- **OFF:** Das Token enthält nur die Rollen, die dem Client explizit über Scopes zugewiesen wurden.




```
{
  "typ": "Bearer",
  "resource_access": {
    "denkmal": {
      "roles": [
        "ratingen_r",
        "duesseldorf_r",
        "read::denkmal",
      ]
    }
  },
  "aud": [ "https://lv.gistest.nrw.de/ldproxy" ],
  "scope": "openid email profile read::denkmal denkmal_r",
  "email_verified": true,
}
```


https://www.keycloak.org/docs/latest/server_admin/index.html#_role_scope_mappings

ldproxy

Konfiguration

- **Öffentliche Berechtigungsgruppe:** Ohne explizite Authentifizierung bzw. entsprechende Berechtigungen ist kein Zugriff auf die API möglich (public: noaccess).
- **Audience:** Es werden nur Token akzeptiert, die spezifisch für die OAPI ausgestellt wurden.
- **Scopes:** Keine Prüfung auf spezifische Scope-Werte im Token. Die Einschränkung erfolgt über die im Token enthaltenen Rollen, deren Zuweisung im Keycloak über Client Scopes gesteuert wird.
- **Permissions:** Jede API liest nur die Rollen aus dem eigenen Namensraum aus.

```yaml
# api.yml
accessControl:
  enabled: true
  groups:
    public: [noaccess]
  audience:
    - https://lv.gistest.nrw.de/ldproxy
  scopes: []
  policies:
    enabled: true
    attributes:
      gemeinde:
        property: gemeinde
    obligations:
      ldproxy:parameters:filter:
        parameter: filter
```

```yaml
# cfg.yml
claims:
  permissions: resource_access.{{apiId}}.roles
```


https://docs.ldproxy.net/de/services/#access-control

Policy Decision Point

AuthzForce

- Bei lesenden Zugriffen auf Features liefert der PDP neben der Entscheidung eine Obligation (Auflage) zurück.
- Diese Auflage enthält einen spezifischen Query-Filter, um die Antwort der OAPI serverseitig auf erlaubte Features einzuschränken.

Herausforderung: Rollen basierte Rechtesets

- Besitzt ein Nutzer mehrere Rollen, werden die einzelnen Filter-Auflagen nicht dynamisch zu einer Gesamt-Query kombiniert.
- Jede mögliche Rollen-Kombination müsste vorab statisch definiert werden.

```
<Rule RuleId="rule" Effect="Permit"><Condition><Apply FunctionId="urn:oasis:names:
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
    ratingen_r
  </AttributeValue>
  <AttributeDesignator AttributeId="ldproxy:claim:permissions" Category="urn:oasis
</Apply></Condition></Rule>
<ObligationExpressions><ObligationExpression ObligationId="ldproxy:parameters" Ful
  <AttributeAssignmentExpression AttributeId="ldproxy:parameters:filter">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
      gemeinde = 'Ratingen'
    </AttributeValue>
  </AttributeAssignmentExpression>
</ObligationExpression></ObligationExpressions>

<Rule RuleId="rule" Effect="Permit"><Condition><Apply FunctionId="urn:oasis:names:
  <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
    duesseldorf_r
  </AttributeValue>
  <AttributeDesignator AttributeId="ldproxy:claim:permissions" Category="urn:oasis
</Apply></Condition></Rule>
<ObligationExpressions><ObligationExpression ObligationId="ldproxy:parameters" Ful
  <AttributeAssignmentExpression AttributeId="ldproxy:parameters:filter">
    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
      gemeinde = 'Düsseldorf'
    </AttributeValue>
  </AttributeAssignmentExpression>
</ObligationExpression></ObligationExpressions>

    <AttributeValue DataType="http://www.w3.org/2001/XMLSchema#string">
      gemeinde IN ('Ratingen','Düsseldorf')
    </AttributeValue>
```

Policy Decision Point

Open Policy Agent

- Alternative zur statischen Konfiguration der Rechtesets in XACML > Open Policy Agent.

- Rechtesets werden nicht statisch konfiguriert, sondern als Skripte programmiert.
- Filter für die OGC API werden basierend auf den aktuellen Rollen des Nutzers dynamisch erzeugt.

```
obligations := o if {
  ldproxy.api == "denkmal"
  ldproxy.operation in ["getItems", "getTile"]

  gemeinde_cql := concat("','", gemeinde_read)

  filter_cql := concat("", ["gemeinde IN ('", gemeinde_cql, "')"])

  o := {ldproxy.obligation_parameter("filter"): filter_cql}
}
```

QGIS

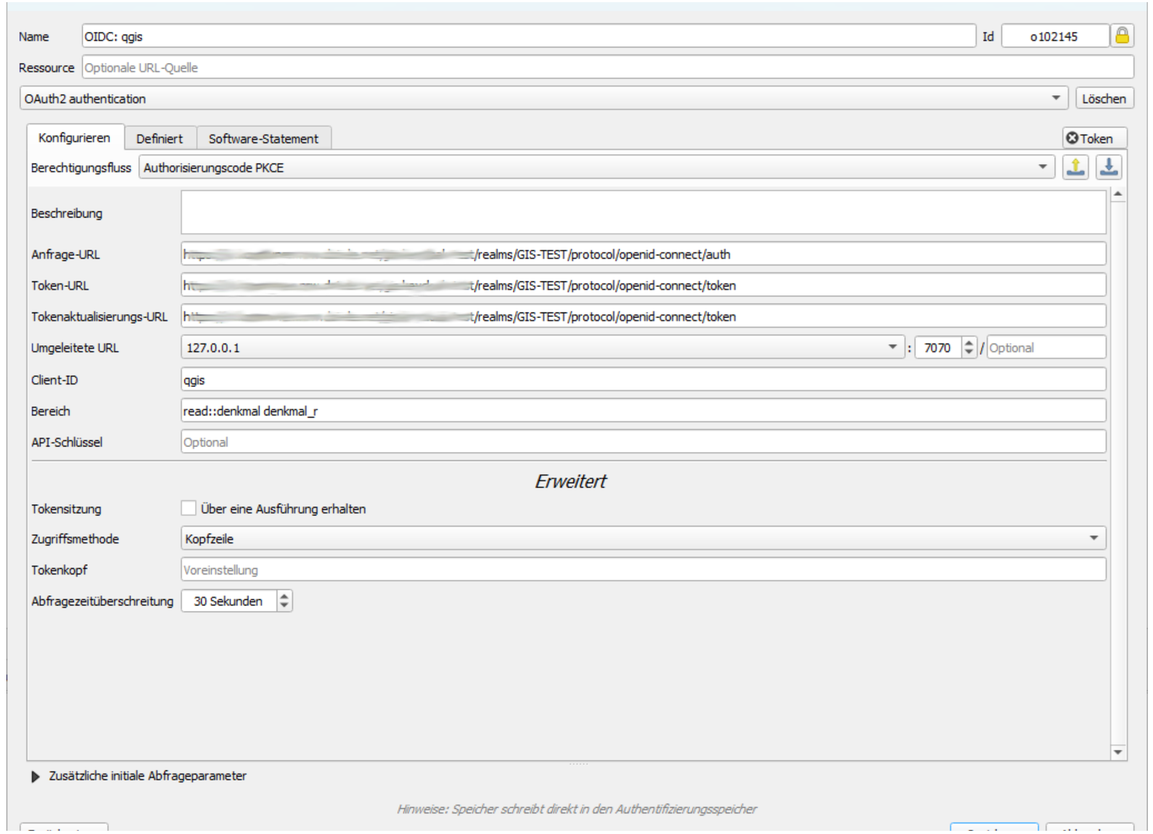
Verfügbare OIDC Authentifizierungs-Methoden:

- Authorization Code Flow + PKCE
- Authorization Code Flow + Client Secret
- Implicit Flow (veraltet)
- Resource Owner Password Credentials Grant (veraltet)

Voraussetzung für alle Flows ist die Registrierung von QGIS als Client im Keycloak.

Technischer Ablauf:

- QGIS startet für den Login kurzzeitig einen lokalen Webserver.
- Die Eingabe der Anmeldedaten erfolgt sicher im Standard-Browser direkt beim Identity Provider (IdP).
- Nach erfolgreichem Login sendet der IdP einen Authorization Code an den lokalen Webserver zurück.



Fazit

- Durch etablierte Standards wie OIDC und XACML lässt sich die OAPI gut in unsere Infrastruktur integrieren.
- Ein wesentlicher Vorteil ist dabei die Unabhängigkeit von einzelnen Herstellern: Das System schreibt keine spezifische Zusatzsoftware vor, sondern setzt auf herstellerneutrale Schnittstellen.

Impressum

Landesbetrieb
Information und Technik Nordrhein-Westfalen

Mauerstraße 51
40476 Düsseldorf
Telefon: 0211 9449-01
Telefax: 0211 9449-8000