Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Departement für
Wirtschaft, Bildung und Forschung WBF

Information Service Center WBF ISCeco

IAM WBF

Factsheet OAuth

Dokumentname	IAM_WBF_-_Factsheet_OAuth v1.2.docx
Version	1.2
Status	Genehmigt zur Nutzung
Author(en)	Michael Wachter (Ext.), Mario Sitz
Ausgabedatum	05.08.2021
Dokument-ID	tbd

Inhalt

1 ALLGEMEINE RESTRIKTIONEN UND ANFORDERUNGEN

1.1 Anbindung

Es wird empfohlen mindestens zwei Registrierungen vorzunehmen (1x Nicht-Produktion und 1x Produktion). Die ClientIDs für Prod und NonProd sind unterschiedlich.

1.2 Unterstützter Token-Typ des Access-Tokens

Das IAM WBF unterstützt nur Access-Tokens vom Token-Typ Bearer.

Der Token-Typ MAC wird nicht unterstützt.

1.3 Unterstützte Flows für Authorization Grant

Bei OAuth2 gibt es 4 Flows, davon werden 2 von IAM WBF unterstützt:

- **Authorization Code Grant** ist geeignet für server-seitige (Web-) Anwendungen. Die Zugangsdaten werden in einer Login-Seite im Browser abgefragt, und sind für die Anwendung nicht ersichtlich.
  - **Confidential Client**: ist eine Client-seitige Anwendung, die in der Lage ist, ein ClientPasswort nach aussen hin geheim zu halten.
  - **Public Client:** ist eine Anwendung, die nicht in der Lage ist, ein Client-Passwort geheim zu halten. Bspw. sind das Mobile- oder Javascript-Anwendungen, die gänzlich im Browser des Benutzers laufen. Bei Public Client muss PCKE implementiert werden.
- **Client Credential Grant** ist geeignet für Situationen, in denen sich die Anwendung selbst authentisieren muss/soll.

Flow	Frontend notwendig	Backend notwendig	User-Interaktion notwendig	Client Secret notwendig
Authorization Code Grant	☑	☑	☑	☑ / ☐ (mit PKCE)
Client Credential Grant	☐	☑	☐	☑

Welcher Flow soll wann eingesetzt werden?

Authorization Code Grant: Zugriff im Namen des Benutzers

Client Credential Grant: Zugriff im Namen eines technischen Accounts

1.4 Gültigkeit der Token

Token	Dauer
Access Token	15Min
RefreshToken	Gültigkeit: 30 Min Sessiondauer max: 10 Stunden Session idle: 30 Min

RefreshToken	Gültigkeit: unbegrenzt
(mit offline_access)	Sessiondauer max: 30 Tage
	Session idle: 30 Tage

AccessToken können durch RefreshToken erneuert werden.

Die RefreshToken können maximal 6 Mal wiederverwendet werden. Wird ein neues RefeshToken benutzt, werden die älteren ungültig.

Die Tokengültigkeit wird auch in der Response von TokenService mitgeliefert:

Response Json:


```
{
  "access_token": "eyJhbGciO....",
  "expires_in": 900,
  "refresh_expires_in": 1800,
  "refresh_token": "eyJhbGciO...",
  "token_type": "bearer",
  "id_token": "eyJhbGciO....",
  "not-before-policy": 1593430363,
  "session_state": "286d33db-5521-40e1-8c5d-2496f8902d79",
  "scope": "openid extId animaltracing"
}
```


Response Json mit offline Access:


```
{
  "access_token": "eyJhbGciO...",
  "expires_in": 900,
  "refresh_expires_in": 0,
  "refresh_token": "eyJhbGciO...",
  "token_type": "bearer",
  "id_token": "eyJhbGciO...",
  "not-before-policy": 1593430363,
  "session_state": "286d33db-5521-40e1-8c5d-2496f8902d79",
  "scope": "openid offline_access extId animaltracing"
}
```


2 REGISTRIERUNG

Für die Registrierung müssen 3 Einstellungen vorgenommen werden:

1. Setup für den Zugriff festlegen, d.h. Bestehendes auswählen oder Neues anlegen.
2. Client Entity anlegen
3. Resource Server Entity anlegen / anpassen

2.1 Setup

Ein Setup beinhaltet folgende Punkte:

Eintrag	Beschreibung
Setup Name	Der Name des Setups
Default Scope Policies	Für jeden Flow wird festgelegt, ob Scopes erlaubt sind, und ob bzw. wie häufig eine Zustimmung durch den Benutzer erforderlich ist.
Default Client Policies	Legt fest welche Flows und Feature zulässig sind.
TTL	Time-to-Live Einstellungen für die verschiedenen Token und den Persistierung der Zustimmung

Ein Setup darf wiederverwendet werden.

2.2 Scopes

Im Normalfall werden die benötigten Scopes bereits bei der Konfiguration des Clients festgelegt. Somit müssen sie nicht angefragt werden.

Es werden nur die definierten Scopes zurückgeliefert.

2.2.1 Benutzer Scopes

Nicht alle Benutzerattribute sind im AccessToken enthalten, diese können über den Userinfo Endpunkt bezogen werden.

- address: JSON-object mit folgendem Inhalt:
  - country: Land
  - formatted: komplette Adresse, formatiert mit “\n”
  - locality: Ort
  - postal_code: Postleitzahl
  - street_address: Strasse und Hausnummer
- email: E-mail
- extId: Agate extId
- family_name: Nachname
- given_name: Vorname
- KT_ID_P: kantonale Benutzer ID
- locale: Kontaktsprache, z.B. «de»
- loginid: Agatenummer
- phone: Telefonnummer
- roles: Rollen
- title: Anrede

2.2.2 SAML Service

Für Applikationen, die kein OAuth unterstützen, ist es möglich, aus dem AccessToken ein SAML-Token generieren zu lassen.

- animaltracing: Zugriff mit SAML auf Animaltracing (OAuth Tokens sind bevorzugt)

- ebd: Zugriff mit SAML eBD
- asan: Zugriff mit SAML auf ASAN

2.2.3 Spezial Scopes

IAM WBF unterstützt folgende Spezial-Scopes:

- openid: benötigt für OpenID Connect requests.
- offline_access: fordert ein «refresh_token» an, das länger benutzt werden kann, ohne dass sich der Benutzer neu anmelden muss.

2.3 Client Entity

Ein Client ist eine Anwendung, die auf Ressourcen zugreifen möchte und dabei für die Authentisierung OAuth verwendet. Um einen Client zu erfassen sind folgende Angaben relevant:

Eintrag	Beschreibung
Entity Name	IAM WBF interner Name der Client Entity
Role	Client
Client Identifier	Username der Client Entity
Client Secret	Password der Client Entity
Client URL	Optional URL der Homepage zur Anzeige bei der Zustimmung
Redirect URLs	Absolute URIs der Redirection Endpoints, welche vom Client verwendet werden.
Registered Scopes	Teilmenge der Scopes der Resource Server Entity, welche von der Client Entity genutzt werden dürfen.
Policies	Pro Flow und Feature wird festgelegt, wer die Policy setzt bzw. nicht.
Response Types	Zulässige Flows festlegen
Type	Public vs Confidential
Contacts	Liste von Email-Adresse mit Komma getrennt
TTL	Siehe oben
Validity Date	Datum ab dem die Änderung gültig werden soll

2.4 URIs

2.4.1 "https" Scheme URI Redirection

Einige Betriebssysteme erlauben die Registrierung von «https» URLs. Der Browser leitet die Antwort an die App weiter.

Beispiel:

https://app.example.com/oauth2redirect/example-provider

2.4.2 Loopback Interface Redirection

Falls Systeme das Loopback-Interface nutzen können, darf es verwendet werden. Der Zugriff ist via http/https.

Beispiel:

“http://127.0.0.1:{port}/{path}" für IPv4, und

"https://[::1]:{port}/{path}" für IPv6.

2.4.3 Private-Use URI Scheme Redirection

Bei mobilen Anwendungen für inter-app Kommunikation verwendbar.

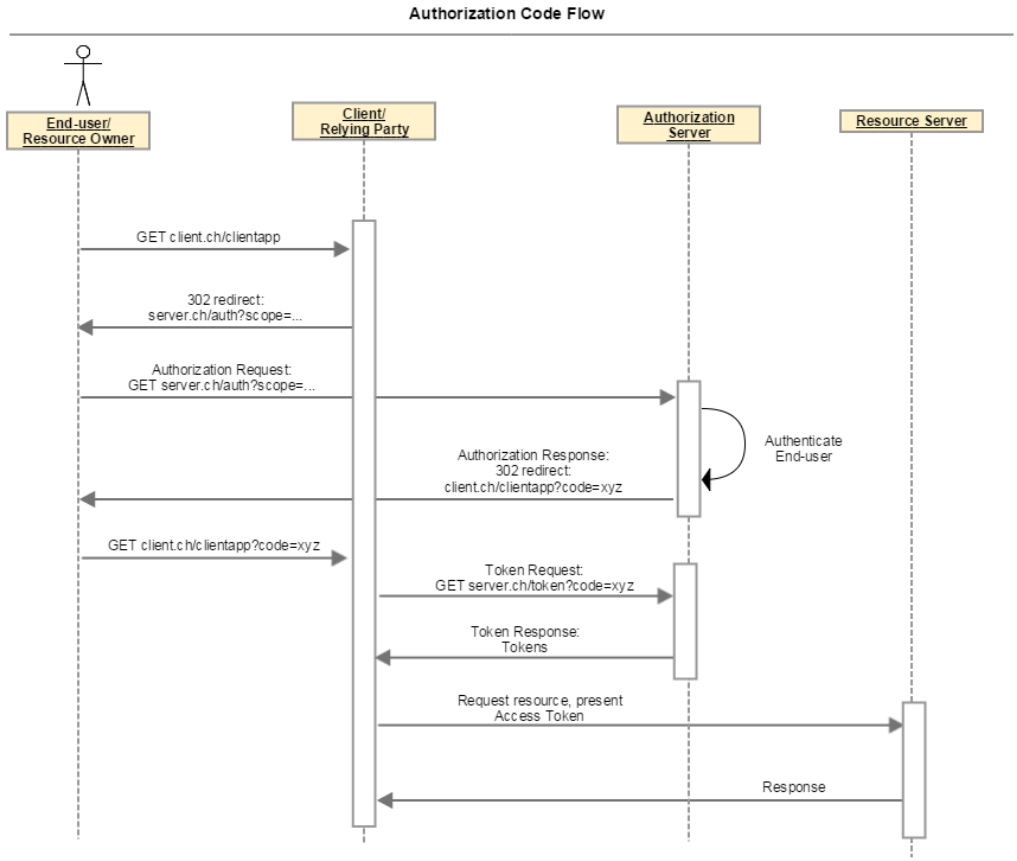
Beispiel:

com.example.app:/oauth2redirect/example-provider

3 FLOW BEISPIELE

3.1 Authorization Code Grant

Der Autorization Code Grant für public Clients soll um die Proof Key for Code Exchange (PKCE) erweitert werden.



3.1.1 Authorization Code beziehen

Aufruf für einen Authorization Code (response_type=code):

```
GET https://server.example.com/authorize?
  response_type=code&
```

```
  client_id=s6BhdRkqt3&
  state=af0ifjsldkj&
  nonce=3f7011bd-fc75-493b-89e9-bace2b01bf41&
  redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
```

Aufruf mit PKCE

```
GET https://server.example.com/authorize?
  response_type=code&
  client_id=s6BhdRkqt3&
  state=af0ifjsldkj&
  nonce=3f7011bd-fc75-493b-89e9-bace2b01bf41&
  code_challenge=CODE_CHALLENGE&
  code_challenge_method=S256&
  redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
```

3.1.2 Access Token mit Authorization Code beziehen

Mit dem Authorization Code (grant_type=authroization_code und code=…) kann ein Access Token bezogen werden.

```
POST https://server.example.com/token
Content-Type: application/x-www-form-urlencoded
  grant_type=authorization_code
  code=SplxlOBeZQQYbYS6WxSbIA
  client_id=s6BhdRkqt3
  client_secret=7FjfpOZBr1KtDRbnfVdmIw
  redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
```

Aufruf mit PKCE:

```
POST https://server.example.com/token
Content-Type: application/x-www-form-urlencoded
  grant_type=authorization_code
  code=SplxlOBeZQQYbYS6WxSbIA
  client_id=s6BhdRkqt3
  code_verifier=YOUR_GENERATED_CODE_VERIFIER
  redirect_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
```

Alternativer Aufruf mit Basic Authentication (base64-kodierte Kombination von client_id und client_secret getrent mit ":")

```
POST https://server.example.com/token
Content-Type: application/x-www-form-urlencoded
Authorization: Basic czZCaGRSa3FOMzo3RmpmcDBaQnIxS3REUmJuZlZkbUl3
  grant_type=authorization_code&
  code=SplxlOBeZQQYbYS6WxSbIA&
  redirect_uri=https%3A%2F%2Fclient.example.com%2Fcb
```

3.1.3 RefreshToken beziehen

Dient zum Erneuern der Accesstoken ohne Neuauthentifizierung durch den Benutzer.

```
POST https://server.example.com/token
```

```
Content-Type: application/x-www-form-urlencoded
  grant_type=refresh_token&
  client_id=s6BhdRkqt3&
  refresh_token=REFRESH_TOKEN
```

3.2 Client Credential Grant

Aufruf für ein Access Token mit den Client credentials (grant_type=client_credentials)

```
POST https://server.example.com/token
Content-Type: application/x-www-form-urlencoded
   grant_type=client_credentials&
   scopes=saml animaltracing&
   client_id=s6BhdRkqt3&
   client_secret=7FjfpOZBr1KtDRbnfVdmIw
```

Alternativer Aufruf mit Basic Authentication (base64-kodierte Kombination von client_id und client_secret getrent mit ":")

```
POST https://server.example.com/token
Authorization: Basic czZCaGRSa3FOMzo3RmpmcDBaQnIxS3REUmJuZlZkbUl3
Content-Type: application/x-www-form-urlencoded
   grant_type=client_credentials&
   scopes=saml animaltracing
```

Beispiel curl:

```
$ curl -X POST -u ${CLIENT_ID}:${CLIENT_SECRET} --data "grant_type=client_credentials" https://idp-rf.agate.ch/auth/realms/agate/protocol/openid-connect/token
```

4 ENDPUNKTE

4.1 Umgebungen

Prod: https://idp.agate.ch

NonProd: https://idp-rf.agate.ch

4.2 Service Endpunkte

Authorization Token Endpoint: /auth/realms/agate/protocol/openid-connect/auth

Token Endpoint: /auth/realms/agate/protocol/openid-connect/token

Token Keys: /auth/realms/agate/protocol/openid-connect/certs

Alle Informationen werden auch über das «Discovery document» bereitgestellt. Dies ist ein JSON mit Informationen über die Authentifizierungsendpunkte und Zertifikate.

- https://idp-rf.agate.ch/auth/realms/agate/.well-known/openid-configuration

- https://idp.agate.ch/auth/realms/agate/.well-known/openid-configuration

4.3 Logout

Die LogoutURI ist ebenfalls im Discovery Dokument enthalten:

/auth/realms/agate/protocol/openid-connect/logout

Es kann der Parameter `post_logout_redirect_uri` mitgegeben werden, dann wird nach einem Logout auf die Applikation zurückgelinkt. Die Redirect URI muss dem System bekannt sein.

Für portalintegrierte System muss die Agate Logout URI verwendet werden:

/logout/

Es kann keine post_logout_redirect_uri mitgegeben werden, es wird zur Portal Seite verlinkt.

5 SERVICES

5.1 SAML Service

Der SAML Service erstellt SAML Tokens auf Basis des AccessTokens. Es muss der Endpunkt des SAMLs im IAM WBF registriert werden.

Die Gültigkeit des SAMLs wird aus dem AccessToken abgeleitet. Es ist nur solange gültig wie das AccessToken.

Response: JSON

{ "SAMLAssertion" : "base64encodedSAML" }

Beispiel:

```
POST https://server.example.com/oauth/api/saml HTTP/1.1
Authorization: Bearer  ACCESSTOKEN
Content-Type: application/x-www-form-urlencoded
  endpointAddress=https://ws-en.wbf.admin.ch/Livestock/AnimalTracing/2
```

mit JSON:

```
POST https://server.example.com/oauth/api/saml HTTP/1.1
Authorization: Bearer  ACCESSTOKEN
Content-Type: application/json
 {"endpointAddress": "https://ws.wbf.admin.ch/Livestock/AnimalTracing/2" }
```

5.2 Userinfo

Die Attribute vom Benutzer können abgefragt werden. Die Scopes werden oben erklärt.

Es werden nur Attribute zurückgegeben, bei denen der Client berechtigt ist.

GET https://idp-rf.agate.ch/auth/realms/agate/protocol/openid-connect/userinfo
Authorization: Bearer ACCESSTOKEN

Response: JSON


```
{
   "sub": "250000001",
   "phone_number": "0444444444",
   "updated_at": 1440747180,
   "email": "janedoe@example.com",
   "address": {
      "country": "Switzerland",
      "formatted": "Addressline1\nAddressline2\nStreet HouseNumber
      DwellingNumber\nBern\n1100\nSwitzerland ",
      "street_address": "Addressline1\nAddressline2\nStreet HouseNumber DwellingNumber",
      “region”: “Bern”,
      "locality": "Bern",
      "postal_code": "3000"
   },
   "name": "Ms. Jane Doe",
   "locale": "en-US",
   "birthdate": "1984-07-19",
   "gender": "female",
   "family_name": "Doe",
   "given_name": "Jane"
}
```