



# Handbuch

Systemadministration

---

OpenId Connect Protokoll für die Anbindung von Clients -  
Spezifikation des Identity Management in aDIS auf der Basis des OpenId  
Connect Protokolls

## INHALT

1.	INFORMATIONEN ZUM DOKUMENT .....	2
2.	BESCHREIBUNG .....	2
3.	ALLGEMEINTE INFORMATIONEN .....	3
3.1.	ARTEN DER AUTHENTIFIZIERUNG .....	3
3.2.	ADIS/OP ENDPUNKTE .....	3
3.3.	SICHERHEIT DES PROTOKOLLS .....	3
3.4.	ALLGEMEINE KONFIGURATIONSDATEN VOM CLIENT .....	3
3.5.	CLIENT UND ADIS/OP KOMMUNIKATION IN ABSTRAKTER FORM.....	4
4.	DAS PROTOKOLL .....	4
4.1.	DER AUTHENTIFIZIERUNGS-ENDPUNKT .....	4
4.1.1.	Authentifizierung im Autorisierungs-Code Ablauf .....	4
4.1.2.	Authentifizierung im impliziten Ablauf .....	6
4.2.	DER TOKEN-ENDPUNKT .....	7
4.3.	DER USERINFO-ENDPUNKT .....	8
4.4.	DER JSON WEB KEY SET (JWKS)-ENDPUNKT .....	9
4.5.	DER END-SESSION-ENDPUNKT .....	10
4.6.	FEHLERBEHANDLUNG .....	11
4.7.	ID TOKEN, SCOPE UND CLAIMS .....	11
4.8.	ACCESS TOKEN.....	12

## 1. INFORMATIONEN ZUM DOKUMENT

Version des Providers	Datum	Verfasser	Änderungen
0.1	22.10.2016	MV	
0.1	13.03.2017	MV	
0.9.5	03.05.2017	MV	Adisbms-Scope erweitert
0.9.7	29.08.2017	MV	End-Session-Endpunkt & SLO
0.9.7	16.01.2018	MV	Lokalisierung des Providers über ui_locales Parameter; JWKS-Request standardisiert

## 2. BESCHREIBUNG

OpenID Connect basiert auf dem OAuth 2.0 (dem Standard für die Autorisierung von API-Zugriffen im Web) und erweitert diesen um Funktionen für Login und Single Sign-on. Die Grundidee von OpenID Connect ist, die Authentifizierung und die Autorisierung des Benutzers in eine eigene Instanz auszulagern, den „OpenID Provider (OP)“, und von dort temporär gültige Tokens zu erhalten (ID Token zur Identifikation und Access Token als Zugriffsberechtigung).

aStec stellt einen OpenID Provider (aDIS/OP) in Form einer Servlet Anwendung bereit, die im Apache Tomcat Container läuft und beim Kunden installiert werden kann. Der OpenID Provider hat zum einen die Aufgabe die Anfragen von den zugelassenen Clients zu bearbeiten. Zum anderen authentifiziert und autorisiert er die Benutzer, die mit den Clients kommunizieren. Die Benutzer melden sich mit ihren Bibliotheksdaten beim OpenID Provider an. Dem Client wird eine vom aDISServer generierte und namensunabhängige ID und Zugriffsberechtigungen herausgegeben. Der OpenID Provider führt Sitzungen über die von Clients angeforderten Nutzerinformationen und -berechtigungen. Der OpenID Provider ist Single Sign-On (SSO) fähig.

Die OpenID Connect Spezifikation befindet sich unter:

[http://openid.net/specs/openid-connect-core-1\\_0.html](http://openid.net/specs/openid-connect-core-1_0.html)

Als **Client** wird ein Lieferant von E-Medien bezeichnet, der die Nutzer über eine Datenbank authentifizieren möchte. Als **aDIS/OP** wird der von aStec entwickelte OpenID Provider bezeichnet.

### 3. ALLGEMEINTE INFORMATIONEN

#### 3.1. Arten der Authentifizierung

Der aDIS/OP unterstützt folgende Authentifizierungsabläufe:

- den Autorisierungs-Code Ablauf und
- den impliziten Ablauf

#### 3.2. aDIS/OP Endpunkte

In aDIS/OP sind folgende Endpunkte implementiert:

- Authentication-Endpunkt: `https://<kundenhost>/oidcp/authorize`
- Token-Endpunkt: `https://<kundenhost>/oidcp/token`
- UserInfo-Endpunkt: `https://<kundenhost>/oidcp/userinfo`
- JSON Web Key Set-Endpunkt: `https://<kundenhost>/oidcp/jwks`
- End-Session-Endpunkt: `https://<kundenhost>/oidcp/logout`

Die in der Spezifikation definierten Discovery und Registry-Endpunkte wurden vorab nicht implementiert.

#### 3.3. Sicherheit des Protokolls

Die Kommunikation mit dem aDIS/OP erfolgt über das HTTPS Protokoll. Die ID Tokens werden immer signiert und können verschlüsselt übermittelt werden.

#### 3.4. Allgemeine Konfigurationsdaten vom Client

Um einen Client an aDIS/OP anzumelden sind folgende Daten vom Client-Betreiber nötig:

Parameter	Beschreibung
<code>client_id</code>	Id des Clients.
<code>client_secret</code>	Client Secret für <code>client_secret_basic</code> -Parameter und für die Signierung via HMAC 256
<code>redirect_uri</code>	Callback URI des Clients.

### 3.5. Client und aDIS/OP Kommunikation in abstrakter Form

- Der Client schickt eine Anfrage an den Autorisierungs-Endpunkt des aDIS/OPs.
- Der OP authentifiziert und autorisiert den Benutzer.
- Der OP schickt eine Antwort mit der Chiffre (Code) bzw. dem ID Token an den Client
- Wenn der Client eine Antwort mit der Chiffre erhält:
  - Der Client schickt eine Anfrage mit der Chiffre an den Token-Endpunkt.
  - Der OP schickt eine Antwort mit dem ID Token an den Client.

## 4. DAS PROTOKOLL

### 4.1. Der Authentifizierungs-Endpunkt

Die Authentifizierungsanfrage muss dem Standard der [http://openid.net/specs/openid-connect-core-1\\_0.html#AuthRequest](http://openid.net/specs/openid-connect-core-1_0.html#AuthRequest) entsprechen. Die Anfrage und die Antwort zum Authentifizierungs-Endpunkt werden über den Browser des Benutzers abgehandelt. Der Client sendet via Umleitung (redirect) eine Anfrage an den aDIS/OP zum Browser. Die Umleitung wird vom Browser in eine GET-Anfrage umgewandelt und an den aDIS/OP gesendet. aDIS/OP validiert die Anfrage und fordert den Benutzer auf, sich über ein Login-Formular anzumelden. (Bei valider Anmeldung kann Benutzer je nach Konfiguration zur Consent-Seite weitergeleitet werden). aDIS/OP sendet eine Antwort via Umleitung des Browsers zurück zum Client.

Pfad zum aDIS/OP Authentifizierungs-Endpunkt:

`oidcp/authorize`

#### 4.1.1. Authentifizierung im Autorisierungs-Code Ablauf

Der Autorisierungs-Code Ablauf ist für die Webanwendungen mit der serverseitigen Komponente vorgesehen. Die Autorisierung erfolgt in 2 Schritten: im ersten Schritt wird ein Nutzer via Browser-Weiterleitung autorisiert und dem Client ein kurzlebiger Autorisierungscode erteilt (am Authentifizierungs-Endpunkt). Im zweiten Schritt wird der Autorisierungscode gegen den ID Token und den Access Token über eine direkte Verbindung zwischen dem Client und dem aDIS/OP ausgetauscht (am Token-Endpunkt).

Parameter für die Authentifizierungsanfrage:

Parameter	Erfordert	Beschreibung
client_id	ja	Id des Clients
redirect_uri	ja	Callback URI des Clients
response_type	ja	„code“
state	ja	Status zum Verifizieren erzeugt vom Client
scope	ja	„openid“ oder „openid adisbms“
claims	nein	libuser_state und/oder libuser_group und/oder libuser_age (Info: Standard-Scopes (wie "profile", "email", ...) und die entsprechenden Standard-Claims werden nicht bedient. Bei Bedarf sind Anpassungen und Erweiterungen möglich.)
prompt	nein	„none“ oder „login“ oder „consent“
ui_locales	nein	Lokalisierung der Provider-Seiten

Authentifizierungsanfrage Beispiel:

```
HTTP/1.1 302 Found
Location: https://<kundenserver>/oidcp/authorize?
  response_type=code
  &client_id=222222
  &redirect_uri=https://myclient/AuthClient/cb
  &scope=openid adisbms
  &state=u3qQh6oD6Cnggbko4x7fMW2l5ZP_ITN7cMzomi0EWcA
  &nonce=pCZhRkUw_pX9QZkK9sdNgUlF7rcPlUta23deID_XG4g
  &prompt=consent
```

Antwort-Angaben der erfolgreichen Authentifizierung:

Parameter	Beschreibung
state	Status zum Verifizieren erzeugt vom Client
code	Code zum Abholen des ID Tokens und des Access Tokens

Antwort Beispiel:

```
HTTP/1.1 302 Found
Location: https://<clienthost>/
    &code =adduefjaowei
    &state=af0ifjsldkj
```

#### 4.1.2. Authentifizierung im impliziten Ablauf

Der implizite Ablauf ist für den Einsatz in den Anwendungen ohne einer serverseitigen Komponente (z.B. Apps auf den mobilen Geräten) vorgesehen. D.h. in den Anwendungen, die die Vertraulichkeit des Kunden-Geheimnisses nicht garantieren können.

Parameter für die Authentifizierungsanfrage:

Parameter	Erfordert	Beschreibung
client_id	ja	Id des Clients
redirect_uri	ja	Callback URI des Clients
response_type	ja	„id_token“ oder „id_token token“
state	ja	Status zum Verifizieren erzeugt vom Client
nonce	ja	Session-Assoziierungswert des Clients zum ID-Token
scope	ja	„openid“ oder „openid adisbms“
claims	nein	libuser_state und/oder libuser_group und/oder libuser_age
prompt	nein	„none“ oder „login“ oder „consent“

Authentifizierungsanfrage Beispiel:

```
HTTP/1.1 302 Found
```

```
Location: https://<kundenserver>/oidcp/authorize?
  response_type=id_token
  &client_id=s6BhdRkqt3
  &redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
  &scope=openid
  &state=af0ifjsldkj
  &nonce=n-0S6_WzA2Mj HTTP/1.1
```

Antwort-Angaben der erfolgreichen Authentifizierung:

Parameter	Beschreibung
state	Status zum Verifizieren erzeugt vom Client
id_token	signierter ID Token Der Scope bzw. die entsprechenden Claims sind im ID Token enthalten
access_token	Access Token falls response_type „id_token token
expires_in	Ablaufzeit des ID Tokens

Antwort Beispiel:

```
HTTP/1.1 302 Found
Location: https://<clienthost>/
&id_token=eyJ0 ... NiJ9.eyJ1c ... I6IjIifX0.DeWt4Qu ... ZXso
&expires_in=3600
&state=af0ifjsldkj
```

## 4.2. Der Token-Endpunkt

Die Tokenanfrage muss dem Standard der [http://openid.net/specs/openid-connect-core-1\\_0.html#TokenRequest](http://openid.net/specs/openid-connect-core-1_0.html#TokenRequest) entsprechen. Die Anfrage und die Antwort zum Token-Endpunkt erfolgt über direkte Verbindung zwischen dem Client und dem aDIS/OP.

Token-Verschlüsselung: z.Zt. keine

Signatur: HMAC 256 oder RSA 256

Pfad zum aDIS/OP Token-Endpunkt:

/oidcp/token



Anfrage-Parameter beim Autorisierungs-Code Ablauf:

Parameter	Beschreibung
Authorization-HEADER	client_id und client_secret im client_secret_basic Header-Parameter z.B. Basic xxxxxxxxxx
grant_type	authorization_code&code=<code>&redirect_uri=<redirect_uri>

Anfrage Beispiel:

```
POST /token HTTP/1.1
Host: <openIdConnectServer>
Content-Type: application/x-www-form-urlencoded
Authorization: Basic xxxxxxxxxx
grant_type=authorization_code&
code=287654789765678976556789&
redirect_uri=<clienthost>
```

Antwort-Parameter auf die Token-Anfrage:

Parameter	Beschreibung
id_token	signierter ID Token im JSON Web Token Format. Der Scope und die entsprechenden Claims sind im ID Token enthalten.
expires_in	Ablaufzeit des ID Tokens
access_token	Access Token

Antwort Beispiel:

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "token_type": "Bearer",
  "expires_in": 3600,
  "id_token": "eyJhbGciOiJIubz5ln..."
}
```

### 4.3. Der UserInfo-Endpunkt

Die UserInfoanfrage muss dem Standard der <http://openid.net/specs/openid-connect->

[core-1\\_o.html#UserInfo](#) entsprechen. Der UserInfo-Endpoint stellt die Benutzerdaten zum gesendeten Access Token zur Verfügung. Die Anfrage und die Antwort erfolgen zwischen dem Client und dem aDIS/OP über direkte Verbindung.

Pfad zum aDIS/OP UserInfo-Endpoint:

`/oidcp/userinfo`

Anfrage-Parameter:

Parameter	Beschreibung
Authorization-HEADER	enthält ein Access Token (mit Scope und Claims) im client_secret_basic Header-Parameter z.B. Basic <access token>

Antwort-Parameter für erfolgreiche Anfrage:

Parameter	Beschreibung
Content-Type HEADER	application/json
Daten	Claims

Antwort Beispiel:

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "libuser_age": "18",
}
```

#### 4.4. Der JSON Web Key Set (JWKS)-Endpoint

Die JWKS-Anfrage ist in der <https://tools.ietf.org/html/rfc7517> beschrieben. Der JWKS-Endpoint liefert die öffentlichen Signierungsschlüssel des aDIS/OPs. Diese Schlüssel werden fürs Verifizieren der digitalen Signatur des ID Tokens und des Access Tokens verwendet. Die Anfrage und die Antwort zum JWKS -Endpoint erfolgen zwischen dem Client und dem aDIS/OP über direkte Verbindung.

Pfad zum aDIS/OP WebKeySet-Endpoint:

`/oidcp/jwks`

Antwort-Parameter:

Parameter	Beschreibung
POST-Data	JSON-File z.B. Basic <access token>

#### 4.5. Der End-Session-Endpunkt

Die Logout-Anfrage ist im Draft "OpenID Connect Front-Channel Logout 1.0" beschrieben (<http://openid.net/specs/openid-connect-frontchannel-1.0.html>). Dabei zeigt eine Relying Party (Client) dem OpenID Provider das Ende ihrer Sitzung mit dem Benutzer an. Wenn der Benutzer zustimmt, setzt dann der OP dessen Status bei sich auf "nicht authentifiziert". Damit ist der Benutzer auch für alle anderen aktiven Clients abgemeldet (Single Logout "SLO"). In der Folge werden alle aktiven Clients vom Logout benachrichtigt, soweit sie sich dafür registriert haben.

In aDIS/OP ist der End-Session-Endpunkt als „Front-Channel Logout“ realisiert. Beim "Front-Channel Logout" wird der Benutzerdialog von der Web-Site des initiiierenden Clients zum "End-Session Endpoint" des OP geleitet. Clients, die mit einem "frontchannel\_logout\_uri" beim OP registriert sind, werden bei Erlaubnis durch den Benutzer via <iframe>-Requests in der Folgeseite über den SLO informiert.

Optional kann der OP den Benutzer danach zum initiiierenden Client zurückleiten. Dazu muss der Client mit einem oder mehreren URLs unter den "post\_logout\_uris"-Metadaten beim OP registriert sein.

Anfrage-Parameter:

Parameter	Beschreibung
state	[optional] Status zum Verifizieren; erzeugt und benötigt vom Client, um den Zustand des Benutzers zwischen der Abmeldung und dem Rückruf des Clients durch den OP beizubehalten. Die Rückrufadresse ist in dem Parameter post_logout_redirect_uri angegeben.
id_token_hint	[empfohlen] Der zuvor durch den OP ausgegebene ID Token als Hinweis auf die aktuelle authentifizierte Sitzung des Endbenutzers.
Post_logout_redirect_uri	post_logout_redirect_uri [optional] URL, auf die der User-Agent des Endbenutzers nach der Abmeldung umgeleitet wird. URL wird unverändert, wie beim OP registriert, verwendet.

OP-Request-Methode Empfehlung:

GET

Clients werden über durchgeführten SLO vom OP benachrichtigt, wenn sie beim OP mit folgendem Parameter registriert sind:

frontchannel\_logout\_uri:[optional] - RP-URL, über die sich die RP ausloggt, wenn es vom OP in einem Iframe gerendert wird. Der OP hängt grundsätzlich an den frontchannel\_logout\_uri folgende Query-Parameter an:

"iss" (Issuer Id, Id des OP)

"sid" (Session Identifier, ein Claim aus dem ID-Token)

Weitere Parameter können von der RP bei der Registrierung definiert sein

Beispiel:

```
https://<provider_url>/oidcp/logout?
id_token_hint=QWEdfewQERq&
state=1234567890
```

#### 4.6. Fehlerbehandlung

Bei fehlgeschlagenen Anfragen an die Endpunkte sendet aDIS/OP Fehler als Antwort-Parameter an den Client zurück.

Antwort-Parameter:

Parameter	Beschreibung
error	ID des Fehlers
error_description	Text zum Fehler
state	Status vom Client

Beispiel:

```
https://<client_url>?
error_description=Access+denied
&state=oWKNQhsfX-rqwLPJ3H2MRHOiZmx0JQ6XeGvybhNqPxo
&error=access_denied
```

#### 4.7. ID Token, Scope und Claims

Der ID Token ist ein JSON Web Token (<https://tools.ietf.org/html/draft-ietf-oauth-json-web-token-32>). Der ID Token wird vom aDIS/OP mithilfe von Geheimnis (secret) signiert.

Die an den Client gelieferten Daten im ID Tokens sind:

- für den Scope „openid“:
  - iss – Id des aDIS/OPs
  - sub – Id des authentifizierten Benutzerkontos ie Benutzer-Id als Hashwert
  - aud – Id des Clients
  - exp – Ablaufzeit des ID Tokens
  - iat – Erzeugungszeit des ID Tokens
  - nonce – Session-Assoziierungswert beim Client
- für den Scope „adisbms“:
  - libuser\_age – Benutzeralter: o-18. Ab 18 Jahren wird immer 18 geliefert
  - libuser\_group - Benutzergruppe
  - libuser\_state – Benutzerstatus
- über die Claims-Anfrage:
  - libuser\_age – Benutzeralter: o-18. Ab 18 Jahren wird immer 18 geliefert
  - libuser\_group - Benutzergruppe
  - libuser\_state – Benutzerstatus

Beispiel eines ID Tokens:

```
{
  "iss": "https://server.example.com",
  "sub": "24400320",
  "aud": "s6BhdRkqt3",
  "exp": 1311281970,
  "iat": 1311280970
}
```

#### 4.8. Access Token

Die Behandlung des Access Tokens entspricht der Spezifikation [http://openid.net/specs/openid-connect-core-1\\_0.html#HybridAccessToken2](http://openid.net/specs/openid-connect-core-1_0.html#HybridAccessToken2). Der Access Token wird in dem UserInfo-Endpunkt verwendet, um diesen gegen die Claims auszutauschen.