



**Tyclipso**

# OpenID Connect (OIDC)

## OAuth 2.0, JWT, JWK










Frank Hönisch

Technologie fördert Kommunikation

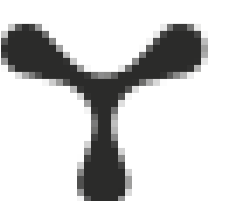
Username

Password

Connect with:

-  Login with Facebook
-  Login with Google
-  Login with Vkontakte
-  Login with Twitter
-  Login with LinkedIn
-  Login with Instagram
-  Login with Amazon
-  Login with Salesforce
-  Login with Microsoft

Remember Me



# Motivation für OAuth und OIDC

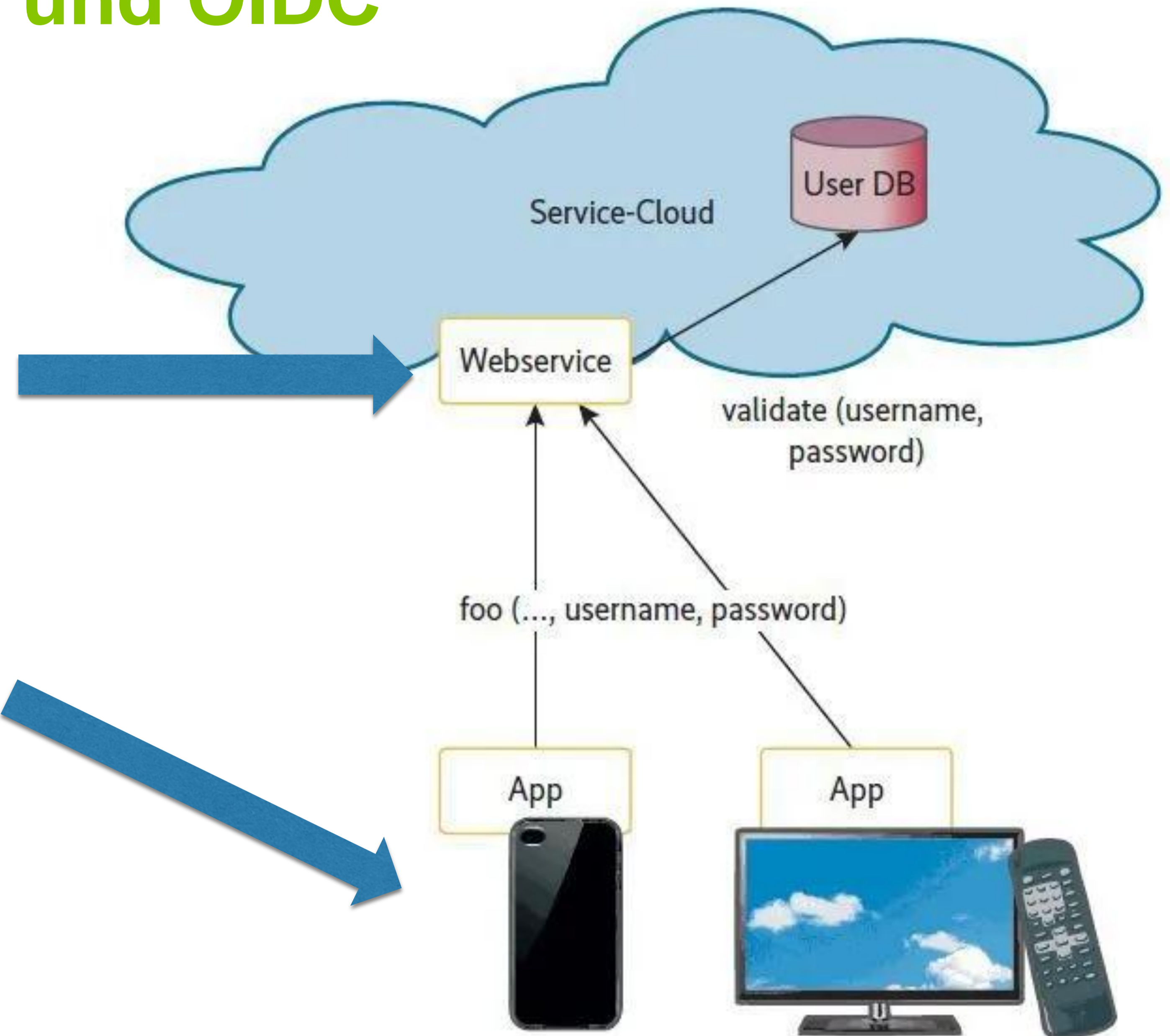
## häufige Architektur

**Authentifizierung  
& Autorisierung**

Username

\*\*\*\*\*

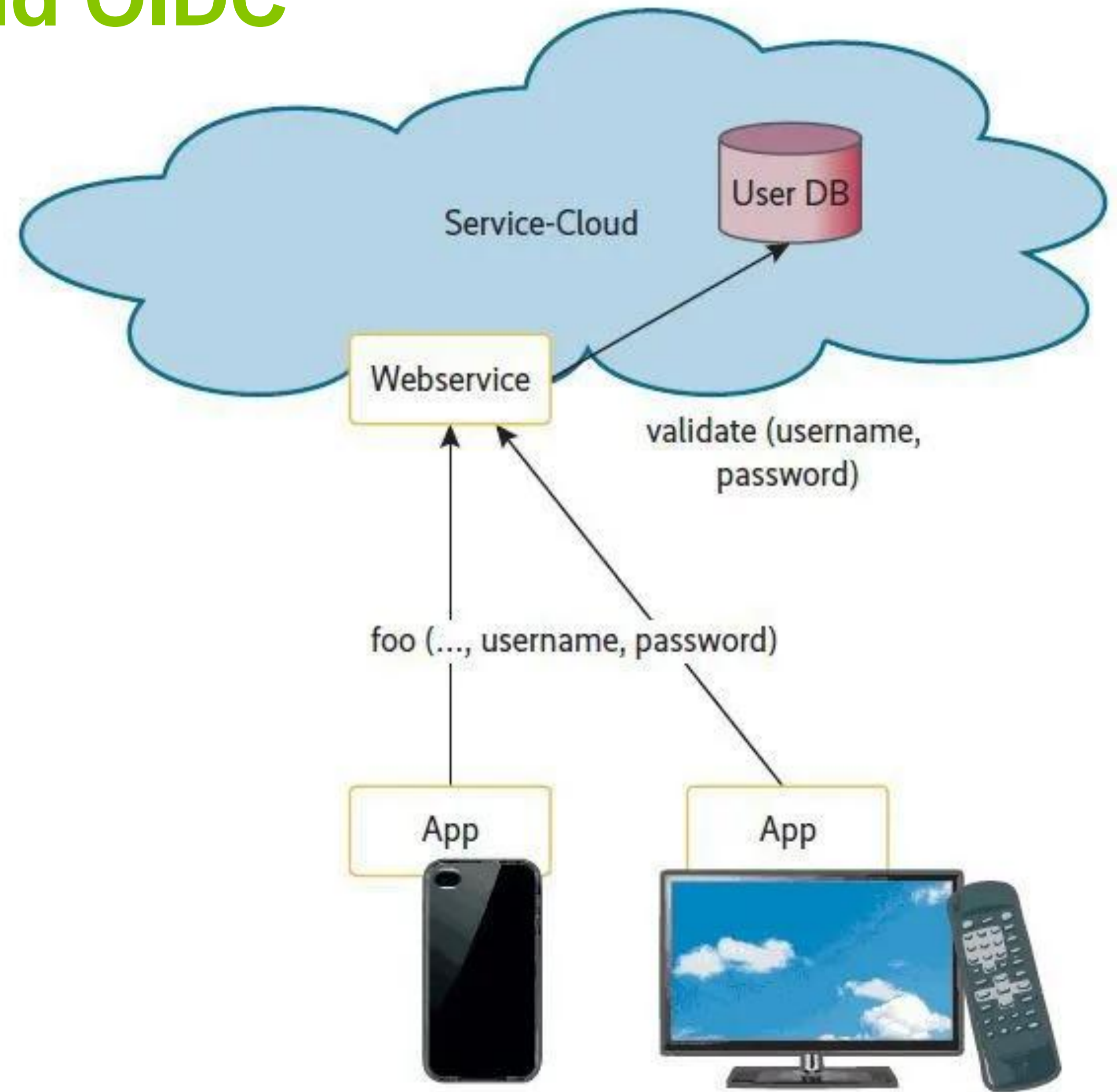
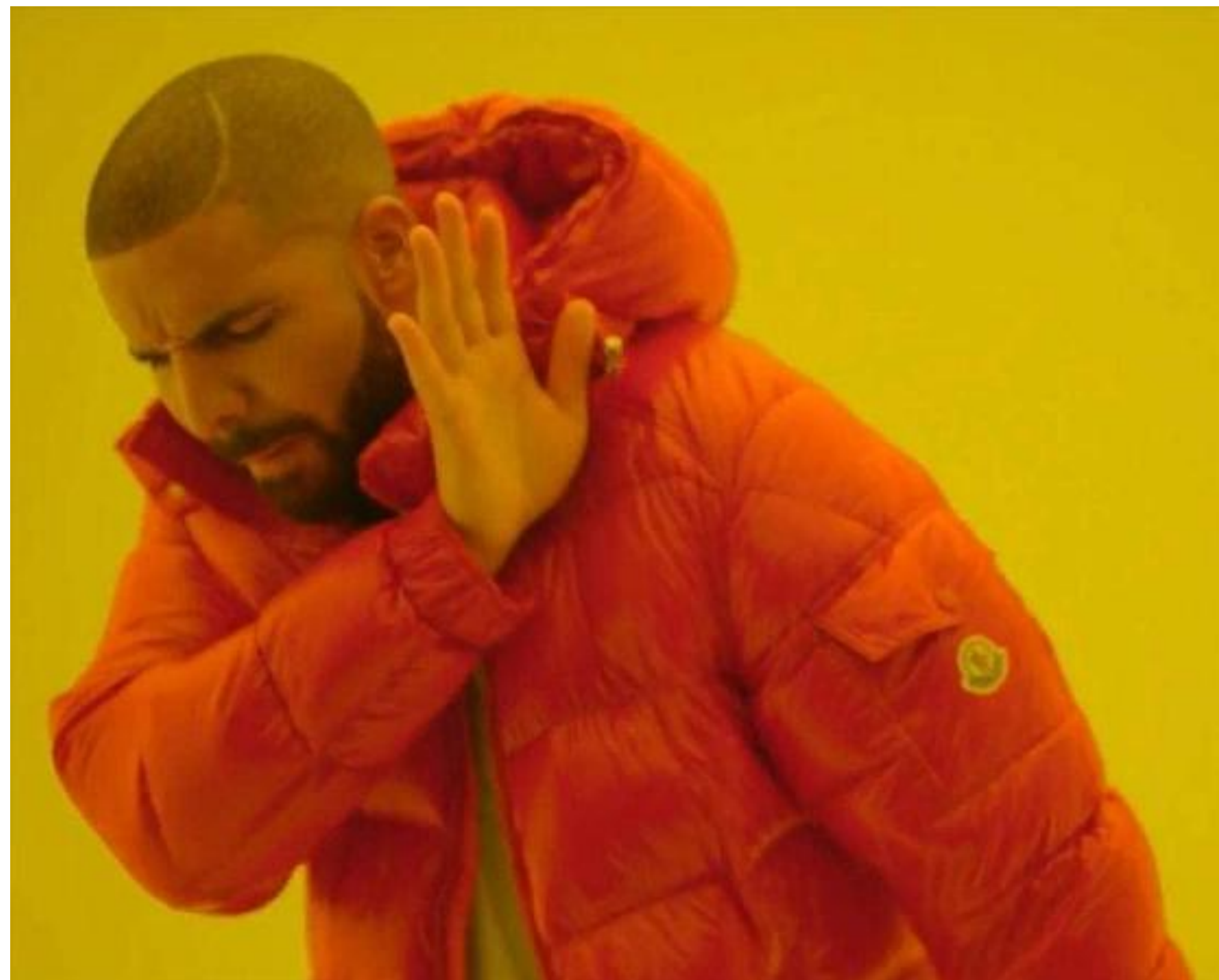
LOGIN



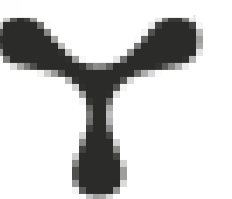
Quelle: <https://www.heise.de/developer/artikel/Flexible-und-sichere-Internetdienste-mit-OAuth-2-0-2068404.html>

# Motivation für OAuth und OIDC

## häufige Architektur



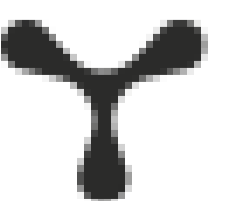
Quelle: <https://www.heise.de/developer/artikel/Flexible-und-sichere-Internetdienste-mit-OAuth-2-0-2068404.html>



# Motivation für OAuth und OIDC

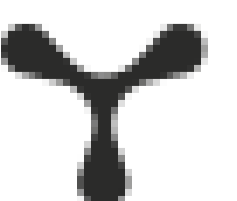


Quelle: <https://www.heise.de/developer/artikel/Flexible-und-sichere-Internetdienste-mit-OAuth-2-0-2068404.html>



# Open Authorization 2.0

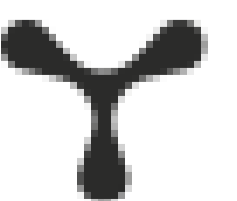
- **Autorisierungsprotokoll**
  - **API-Autorisierung für Apps**
- verwendet **Access Tokens**
  - zur Zugriffskontrolle auf geschützte Ressourcen
  - aber keine Nutzerinformationen



# Open Authorization 2.0

## Liefert keine Informationen:

- Art der Authentifizierung
- Zugangsdaten
- keine Benutzerdaten !!!



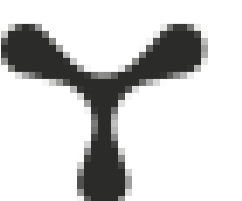
# OpenID Connect

- **Authentifizierungsprotokoll**  
basierend auf OAuth 2.0



Ermöglicht Anwendungen

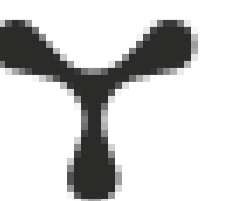
- **Authentifizierung:** Identität eines Nutzers überprüfen
- Liefert grundlegende **Nutzerinformationen**





# Technische Basis

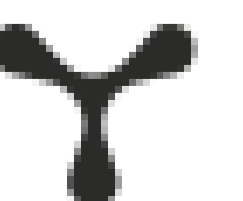
- **HTTP Schnittstelle** mit **REST**-Mechanismen
- Datenformat: **JSON**



# Wie erweitert Open ID Connect OAuth 2.0?

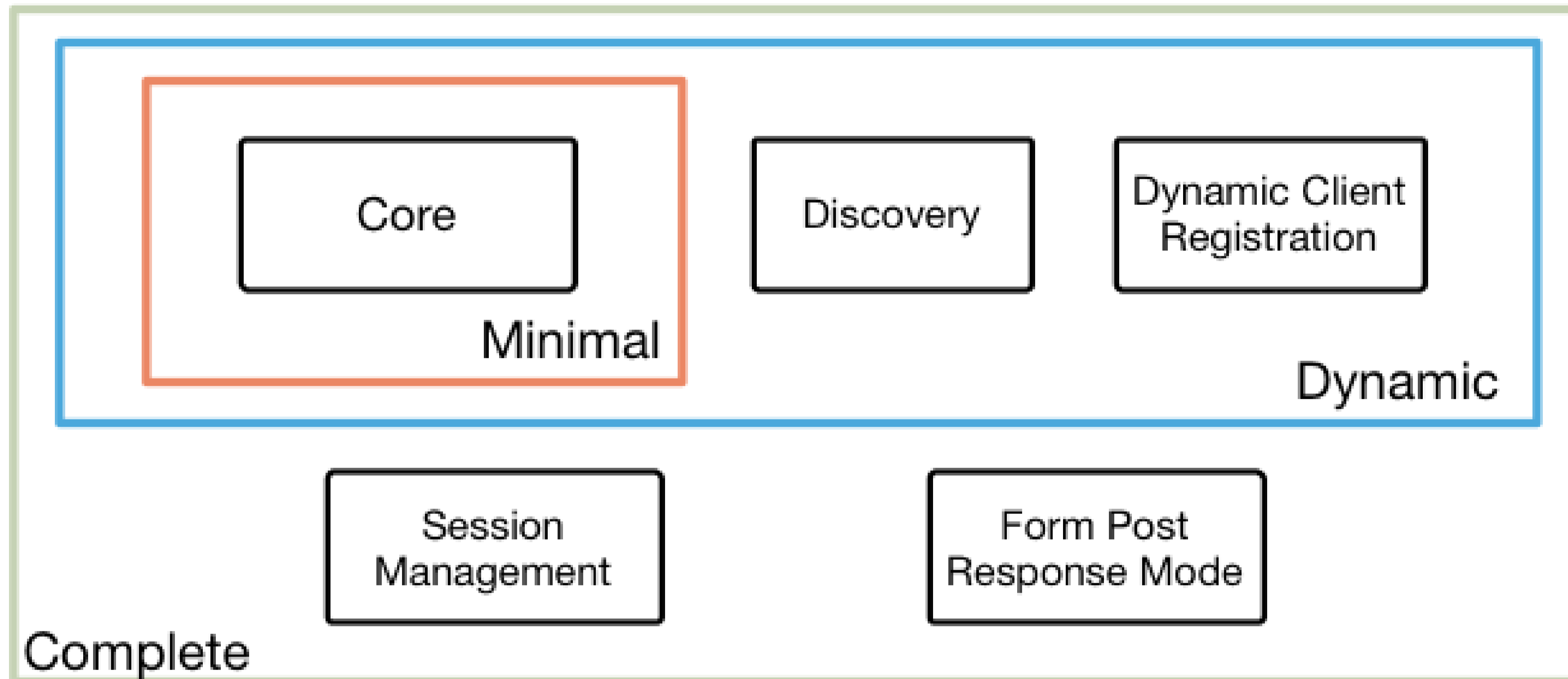
- Open Authorization 2.0 + **Identitätsschicht**
- Informationen zum Benutzer
- ermöglicht Clients die Einrichtung von Anmeldesitzungen

***Open Authorization 2.0 + (Identität, Authentifizierung)***  
***= OpenID Connect***

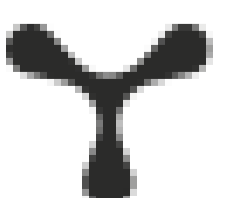
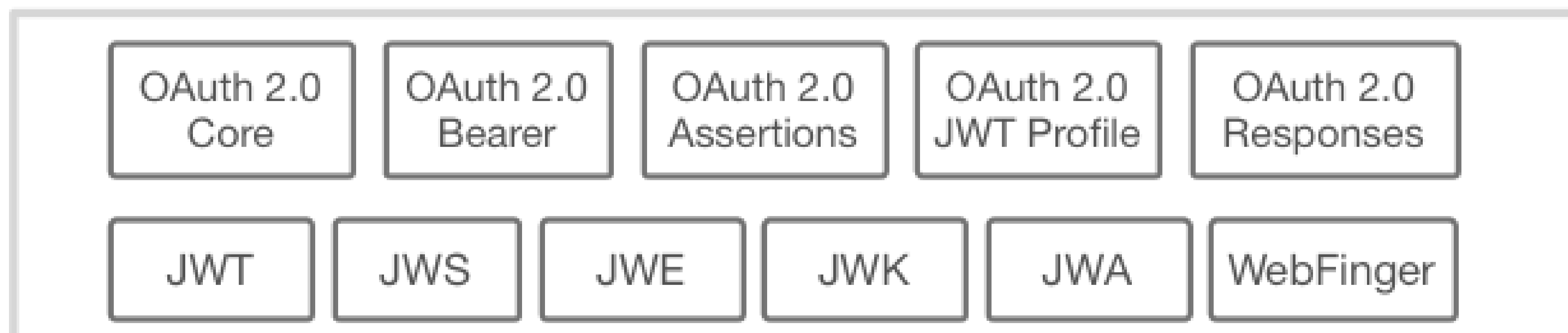


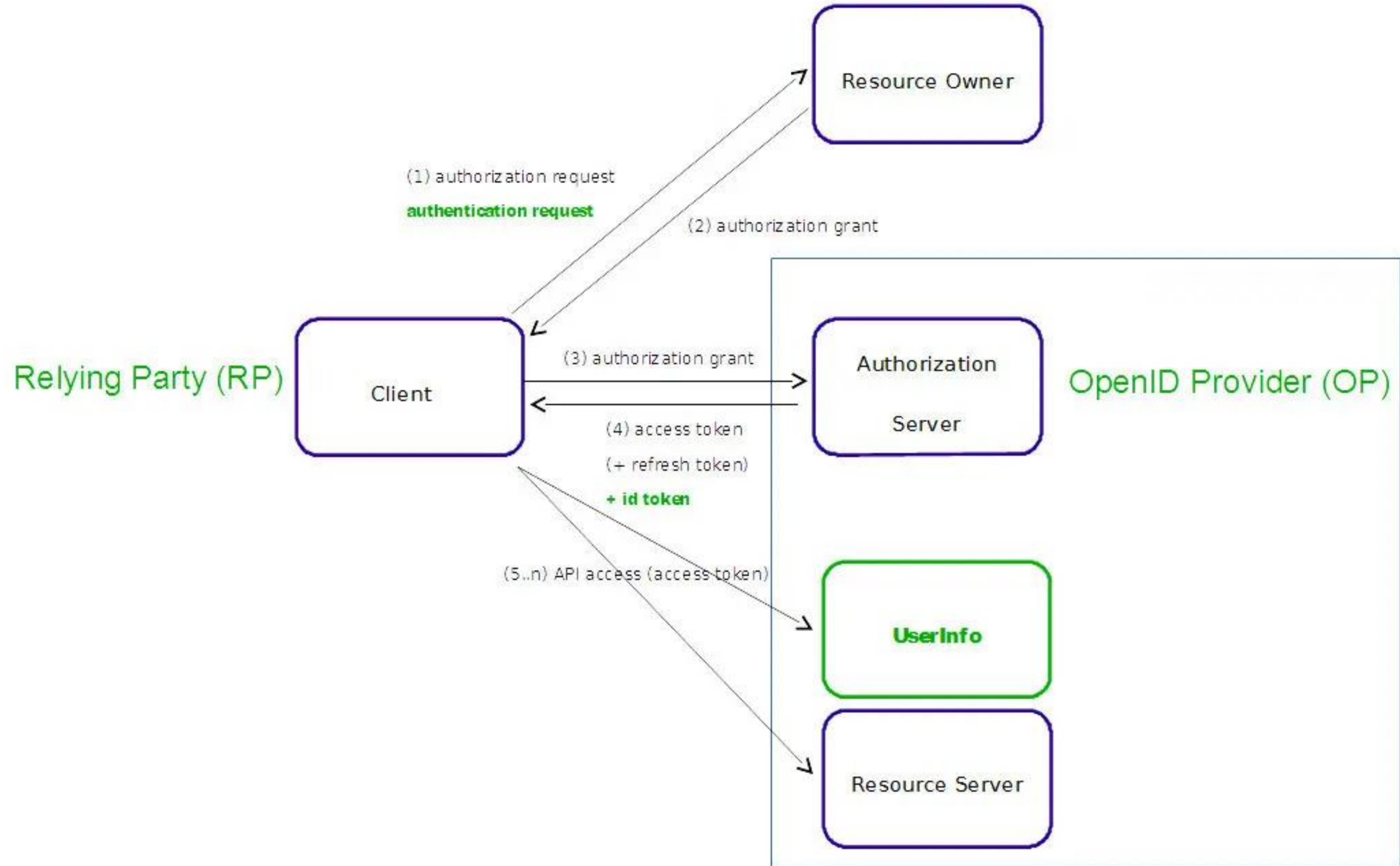
# OpenID Connect Protocol Suite

<http://openid.net/connect>

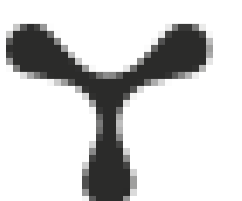


## Underpinnings

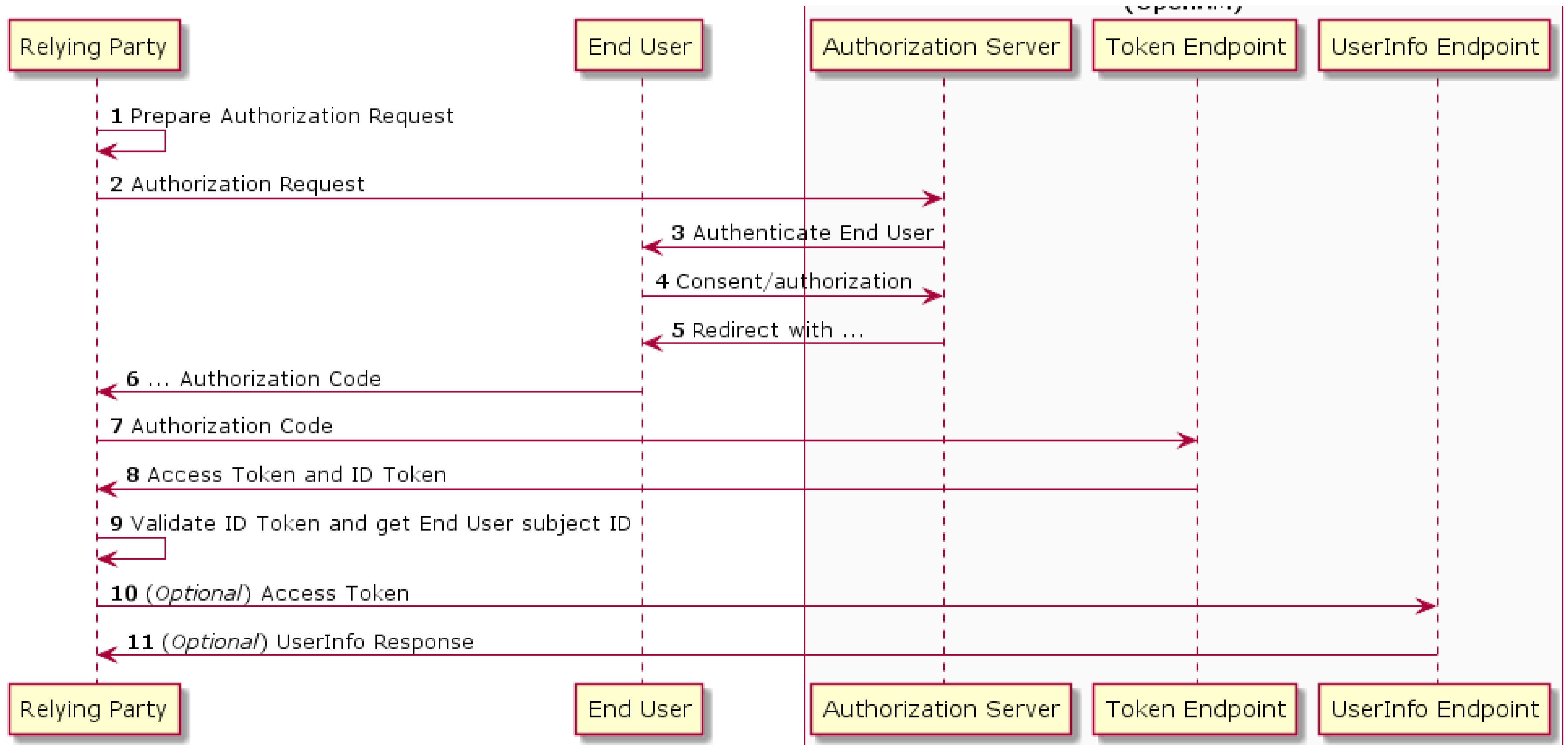




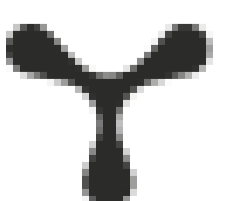
Quelle: <https://www.heise.de/developer/artikel/OpenID-Connect-Login-mit-OAuth-Teil-1-Grundlagen-2218446.html?seite=all>



# Authorization Code Flow



Quelle: <https://backstage.forgerock.com/docs/am/5/oidc1-guide/>

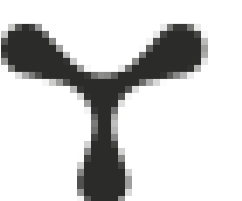


# ID Token = JSON Web Token (JWT)

**Aufbau:** Header, Payload, Signatur

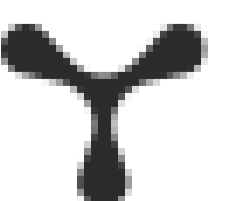
```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MzIyODQyLmVudC50eSMEKKF2QT4fwpMeJf36P0k6yJV_adQssw5c
```

## Codierung

$$jwt = base64UrlEncode(header) + "." + base64UrlEncode(payload) + "." + base64UrlEncode(hash)$$


# ID Token = JSON Web Token (JWT)

| HEADER: ALGORITHM & TOKEN TYPE  |
|---|
| <pre>{<br/>  "alg": "HS256",<br/>  "typ": "JWT"<br/>}</pre>                                     |
| PAYLOAD: DATA   |
| <pre>{<br/>  "sub": "1234567890",<br/>  "name": "John Doe",<br/>  "iat": 1516239022<br/>}</pre> |



# ID Token = JSON Web Token (JWT)

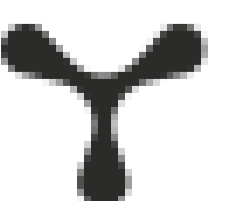
## Signatur

```
var encodedString = base64UrlEncode(header) + "." + base64UrlEncode(payload);
```

```
var hash = RS256(encodedString, secret);
```

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "typ": "JWT",  
  "kid": "c7vvFmXT3W8MkkUKe1wk0BdPyaM=",  
  "alg": "RS256"  
}
```



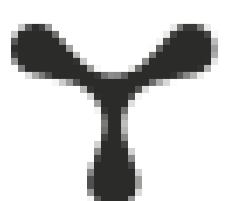


# OIDC Discovery

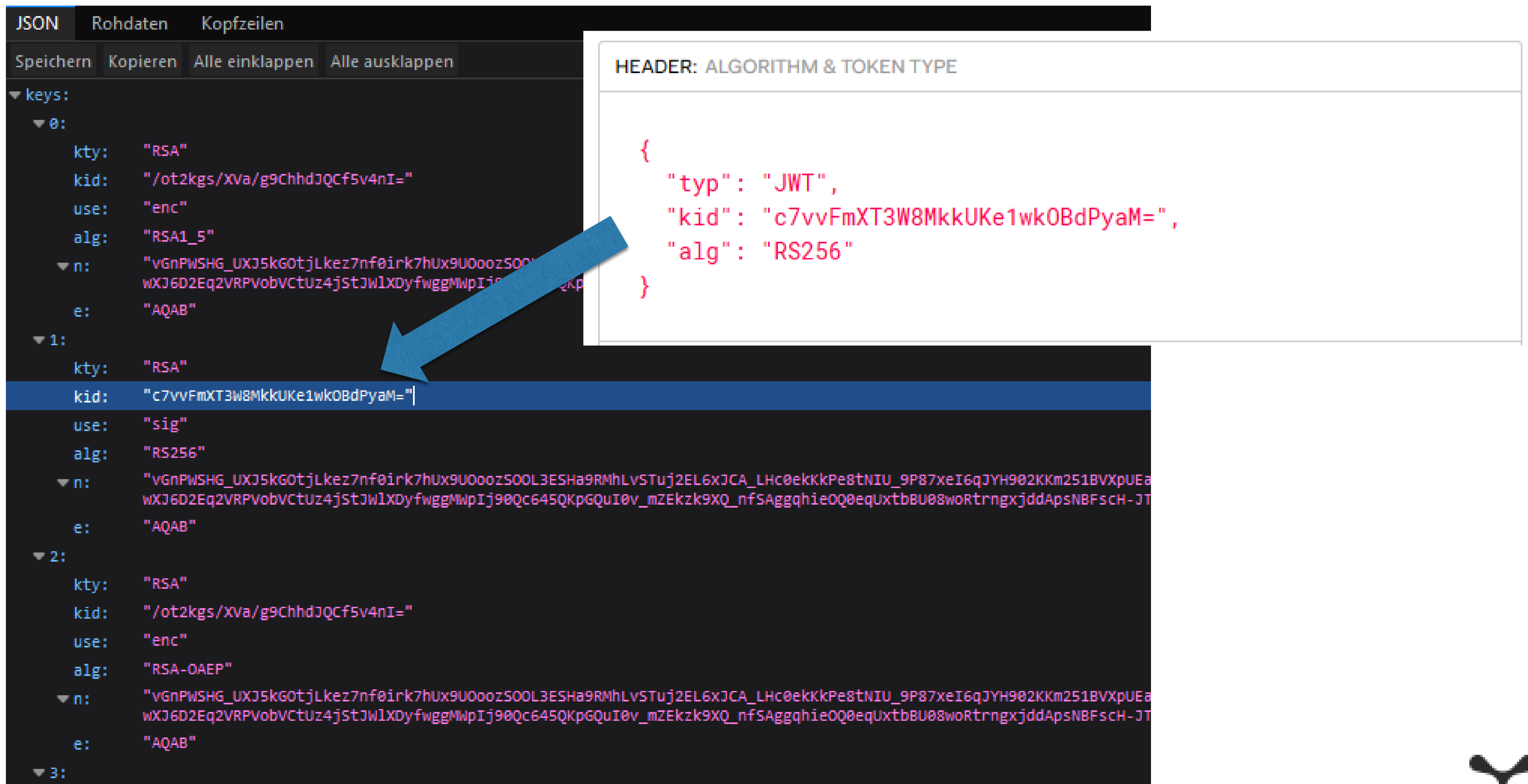
## Metadaten des OpenID-Servers

<https://server.com/.well-known/openid-configuration>

```
issuer: "https://login.int.swissid.ch:443/idp/oauth2"
id_token_encryption_enc_values_supported: [-]
acr_values_supported: [-]
authorization_endpoint: "https://login.int.swissid.ch:443/idp/oauth2/authorize"
request_object_encryption_enc_values_supported: [-]
rsc_request_encryption_alg_values_supported: [-]
claims_supported: [-]
rsc_request_signing_alg_values_supported: [-]
token_endpoint_auth_methods_supported: [-]
token_endpoint: "https://login.int.swissid.ch:443/idp/oauth2/access_token"
response_types_supported: [-]
request_uri_parameter_supported: true
rsc_response_encryption_enc_values_supported: [-]
end_session_endpoint: "https://login.int.swissid.ch:443/idp/oauth2/connect/endSession"
rsc_request_encryption_enc_values_supported: [-]
version: "3.0"
rsc_response_encryption_alg_values_supported: [-]
userinfo_endpoint: "https://login.int.swissid.ch:443/idp/oauth2/userinfo"
id_token_encryption_alg_values_supported: [-]
jwks_uri: "https://login.int.swissid.ch:443/idp/oauth2/connect/jwk_uri"
subject_types_supported: [-]
id_token_signing_alg_values_supported:
  0: "ES384"
  1: "RS384"
```



# OIDC Discovery: Signaturprüfung mit JWK

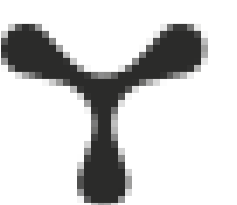


The image shows a JSON viewer interface with a dark theme. The main panel displays a list of keys under the 'keys:' root. Key 1 is highlighted in blue, showing the following properties:

```
key: 1
  kty: "RSA"
  kid: "c7vvFmXT3W8MkkUKe1wk0BdPyaM="
  use: "sig"
  alg: "RS256"
  n: "vGnPWSHG_UXJ5kG0tjLkez7nf0irk7hUx9U0oozS00L3ESha9RMhLvSTuj2EL6xJCA_LHc0ekKkPe8tNIU_9P87xeI6qJYH902KKm251BVXpUEaWxJ6D2Eq2VRPVobVctUz4jstJWlXdyfwggMwpIj90Qc645QKpGQuI0v_mZEKzk9XQ_nfsAggqhieOQ0eqUxtbBU08woRtrngxjddApsNBFsch-JT"
  e: "AQAB"
```

A blue arrow points from the 'kid' value in the JSON to a callout box titled 'HEADER: ALGORITHM & TOKEN TYPE'. The callout box contains the following JSON:

```
{
  "typ": "JWT",
  "kid": "c7vvFmXT3W8MkkUKe1wk0BdPyaM=",
  "alg": "RS256"
}
```



# PHP Implementierungen

## **OpenID Connect-Client**

<https://bitbucket.org/PEOFIAMP/phpoidc>

<https://github.com/jumbojett/OpenID-Connect-PHP>

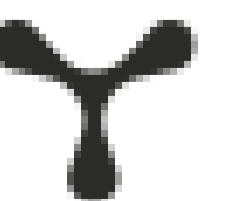
## **OAuth-Client**

<http://oauth2-client.thephpleague.com/>

## **JWT**

<https://github.com/fproject/php-jwt> erweitert

<https://github.com/firebase/php-jwt>



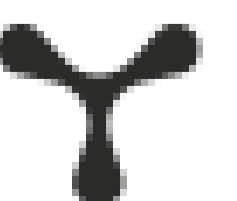
## Referenzen / Artikel

### OpenID Connect

- <https://openid.net/connect/>
- <https://www.heise.de/developer/artikel/OpenID-Connect-Login-mit-OAuth-Teil-1-Grundlagen-2218446.html>
- <https://www.heise.de/developer/artikel/OpenID-Connect-Login-mit-OAuth-Teil-2-Identity-Federation-und-fortgeschrittene-Themen-2266017.html>

### OAuth

- <https://www.heise.de/developer/artikel/Flexible-und-sichere-Internetdienste-mit-OAuth-2-0-2068404.html>



# Tools

## OpenID Connect

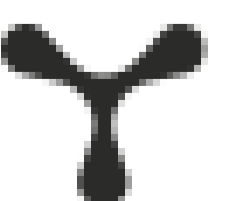
- OpenID Connect Playground: <https://openidconnect.net/>
- OpenID Connect <debugger/>: <https://oidcdebugger.com/>

## OAuth

- OAuth 2.0 <debugger/> : <https://oauthdebugger.com/>

## JWT

- Online JWT decoder / generator: <https://jwt.io>





**Vielen Dank für eure Zeit!**

Grundstraße 1  
01326 Dresden  
**+49 351 3122303**  
office@tyclipso.net