

# A survey of OAuth, OIDC, and Verifiable Credentials (Wallets)

Richard Frovarp

# About

Richard Frovarp

Principal Software Engineer in higher education

Specializing in IAM and federation


# Old Days


<https://blog.codinghorror.com/please-give-us-your-email-password/>


**Are your friends already on Yelp?**


Many of your friends may already be here, now you can find out. Just log in and we'll display all your contacts, and you can select which ones to invite! And don't worry, we don't keep your email password or your friends' addresses. We loathe spam, too.

Your Email Service

☐  Hotmail

☐ 

☐ 

☒ 

Your Email Address

(e.g. bob@gmail.com)

Your Gmail Password

(The password you use to log into your Gmail email)

[Skip this step](#)

# OAuth 2.0

Authorization protocol

Mechanism to grant access to a resource

Several authorization grants (flows)

# Internal Use

- All of your apps can have the same grants
- Or set of grants
- No need to update each app as security for login changes

# Bilateral operation

Need to register your application with the service:

- Name + website
- Redirect URI
- Provided with client ID and client secret

# OAuth 2.0 Roles

- **Resource Owner:** User or system that owns the data
- **Client:** The software accessing the resource
- **Authorization Server:** Issues access tokens on behalf of the Resource Owner
- **Resource Server:** Server that is protecting the Resource Owners data

# OAuth 2.0 Scopes

Defined per resource



# Access Token

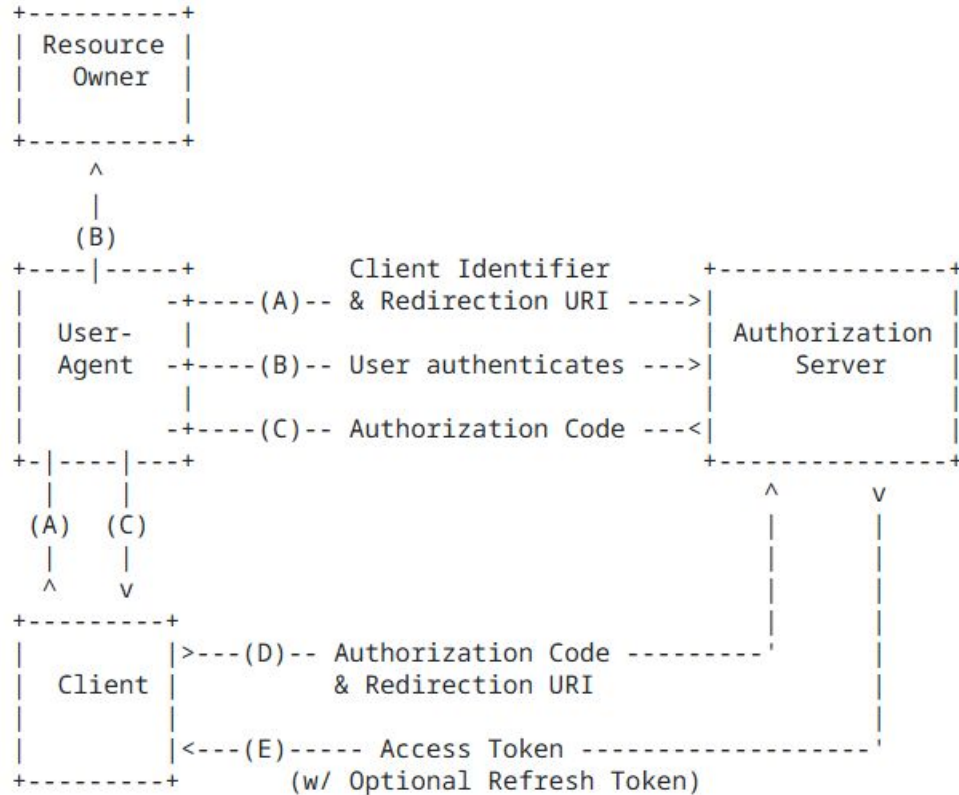
- The token presented by the Client to the Resource Server
- Can be anything
  - Needs to not be guessable when effectively representing a session key
  - Can be a JWT, or anything else

```
{  
  "access_token": "2YotnFZFEjr1zCsicMWpAA",  
  "token_type": "example",  
  "expires_in": 3600,  
  "refresh_token": "tGzv3J0kF0XG5Qx2TlKWIA",  
  "example_parameter": "example_value"  
}
```

# Grant Types

- **Authorization Code Grant:** Traditional grant use by traditional web applications
- **Implicit grant:** Deprecated
- **Authorization Code Grant with PKCE:** Additional steps to make it secure for SPAs and native apps
- **Resource Owner Credentials Grant Type:** requires Client to have Owners creds. So requires trust and inability to redirect in Authorization Code grant.
- **Client Credential Grant Type:** Authenticated by client id and secret. Used for non-interactive applications
- **Device Authorization Flow:** Used by input-constrained devices like TVs
- **Refresh Token Grant:** Uses Refresh Token to get a new Access Token

# Authorization Code Grant



# With PKCE

- Proof Key for Code Exchange
- Used when client key can't be kept secret
  - Decompilers exist
- Client has browser send a hash to Authorization Server
- Client on token retrieval send hash and plain

# Refresh Token

- Can be stored in secure storage on a device
- Provides a "seamless" login if main token expired

# OIDC (OpenID Connect)

- SSO (Single Sign On) technology built on top of OAuth
- RP (Relying Party, Client)
- OP (OpenID Provider)
- Requires **openid** scope
- Get a JWT in return

## OIDC Response

```
{
  "access_token": "S1AV32hkKG",
  "token_type": "Bearer",
  "refresh_token": "8xL0xBtZp8",
  "expires_in": 3600,
  "id_token":
    "eyJhbGciOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ.ewogImlzczyI6ICJodHRwOi8vc2VydmVybWV4YW1wbGUuY29tIiwKICJzdWIiOiAiMjQ4MDJgNzYxMDAxIiwKICJhdWQiOiAiciczZCaGRSa3F0MyIsCiAibm9uY2UiOiAibi0wUzZfV3pBMk1qIiwKICJleHAiOiAxMzExMjgxOTcwLAogIm1ldCI6IDEzMTEyODANzAKfQ.ggW8hZ1EuVLuxNuuIJKX_V8a_OMXzR0EHR9R6jgdqrO0F4daGU96Sr_P6qJp6IcmD3HP990bi1PRs-cwh3L0-p146waJ8IhehcwL7F09JdijmBqkvPeB2T9CJNqeGpe-gccMg4vfKjkM8FcGvnzZUN4_KSP0aAp1t0J1zZWgjxqGBYKhI0tX7TpdQyHE5lcMiKPXFIEIQLVq0pc_E2DzL7emopWoaoZTF_m0_N0YzFC6g6EJb0EoRoSK5hoDalrcvRYLSrQAZZKflyuVCyixEoV9GfnQC3_osjzw2PAithfubEEBLuVVk4XUVrWOLrLl0nx7RkKU8NXNHq-rvKMzqg"
}
```

# JWT

- Header: Algorithm and Token type

```
{  
  "alg": "RS256",  
  "kid": "1e9gdk7"  
}
```

- Payload

```
{  
  "iss": "http://server.example.com",  
  "sub": "248289761001",  
  "aud": "s6BhdRkqt3",  
  "nonce": "n-0S6_WzA2Mj",  
  "exp": 1311281970,  
  "iat": 1311280970  
}
```

- Signature



# Can contain many more claims

- Name, given\_name, family\_name, nickname, preferred\_username
- Profiles
- Picture
- website, email, email\_verified
- gender, birthdate
- address

# JWT Signing

## Algorithms

- Symmetric
- Asymmetric

Modern libraries should require declaring which algorithm is used to ensure verification is correct

# OpenID Federation

- Enables multi-lateral federation
- Currently being worked on
  - [OpenID Federation 1.0 - draft 40](#) published October 24, 2024
  - Italian R&E federation leading in implementation details for R&E
- Allows for many to many trust framework
  - like CAF (Canadian Access Federation), or eduGAIN

# Verifiable Credentials

- Also known as Self Sovereign Identity (SSI)
- Wallets contain a lot of stuff
  - Both physical and digital wallets
  - Not all things are the same in either
- [W3C Verifiable Credentials](#)

# Uses

- Digital driver's license
  - California has their own app, plus Apple & Google
    - Custom app has ability to limit release of attributes
      - Sacramento only
  - Accepted by TSA for several states
  - Online verification
- Diplomas, Degrees, Certificates, and micro-credentials
  - Employer needs to verify academic credential
  - Digital Credentials Consortium <https://digitalcredentials.mit.edu/>
- Digital ID & Authentication Council of Canada
  - <https://diacc.ca/>
  - Advocating for use

- Uses OIDC and PKI
- Flow is such that identity / attribute issuer doesn't know credential has been used
- Credentials expire
- Revocation is odd
- A bunch of different wallet technology
- Issue for those that don't have tech
- Trust
  - [The Wallets Are Coming – But Are We Ready for What's Next?](#) - Heather Flanagan
  - [Multilateral Federation: The Solution to the Problem that Identity Wallets Don't Yet Understand They Have](#) - John Bradley

# Parties

- Issuer
  - Issues credential to holder
  - Signs with reference to public key
- Holder
  - Releases credential to Verifier
  - Signs with reference to public key
- Verifier
  - Needs to verify credential by validating signatures using public keys
  - Needs to know to TRUST the Issuer
    - Likely buy a service?

# Doesn't require blockchain

The blockchain used by the North Dakota Information Technology (NDIT) department was originally developed by Seattle's Evernym and operated as the Sovrin network. Software company Avast acquired Evernym in December 2021 and is shutting down that blockchain, so NDIT will have to find a new chain by the fall, Korsmo told Blockworks.