

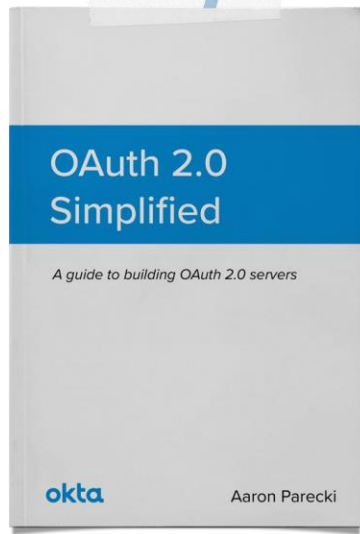
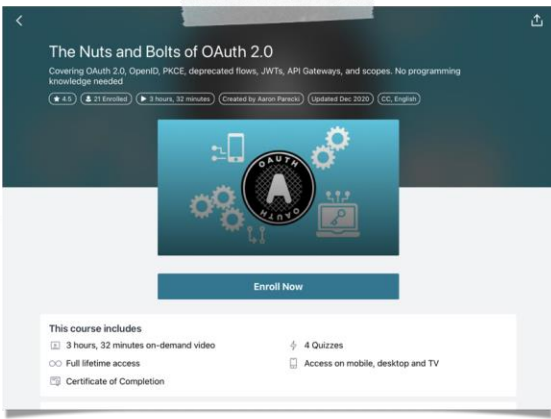
The State of OAuth

Aaron Parecki

Senior Security Architect

Okta

okta



[Docs] [txt|pdf|xml|html] [Tracker] [WG] [Email] [Diff1] [Diff2] [Nits]
Versions: (draft-parecki-oauth-v2-1) 00 01
OAuth Working Group D. Hardt
Internet-Draft SignIn.Org
Intended status: Standards Track A. Parecki
Expires: 5 August 2021 Okta
T. Lodderstedt
yes.com
1 February 2021

The OAuth 2.1 Authorization Framework
draft-ietf-oauth-v2-1-01

Abstract

The OAuth 2.1 authorization framework enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and an authorization service, or by allowing the third-party application to obtain access on its own behalf. This specification replaces and obsoletes the OAuth 2.0 Authorization Framework described in [RFC 6749](#).

[Docs] [txt|pdf|xml|html] [Tracker] [WG] [Email] [Diff1] [Diff2] [Nits]
Versions: (draft-parecki-oauth-browser-based-apps) 00 01 02 03 04 05 06 07
Open Authentication Protocol A. Parecki
Internet-Draft Okta
Intended status: Best Current Practice D. Waite
Expires: April 5, 2021 Ping Identity
October 02, 2020

OAuth 2.0 for Browser-Based Apps
draft-ietf-oauth-browser-based-apps-07

Abstract

This specification details the security considerations and best practices that must be taken into account when developing browser-based applications that use OAuth 2.0.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).



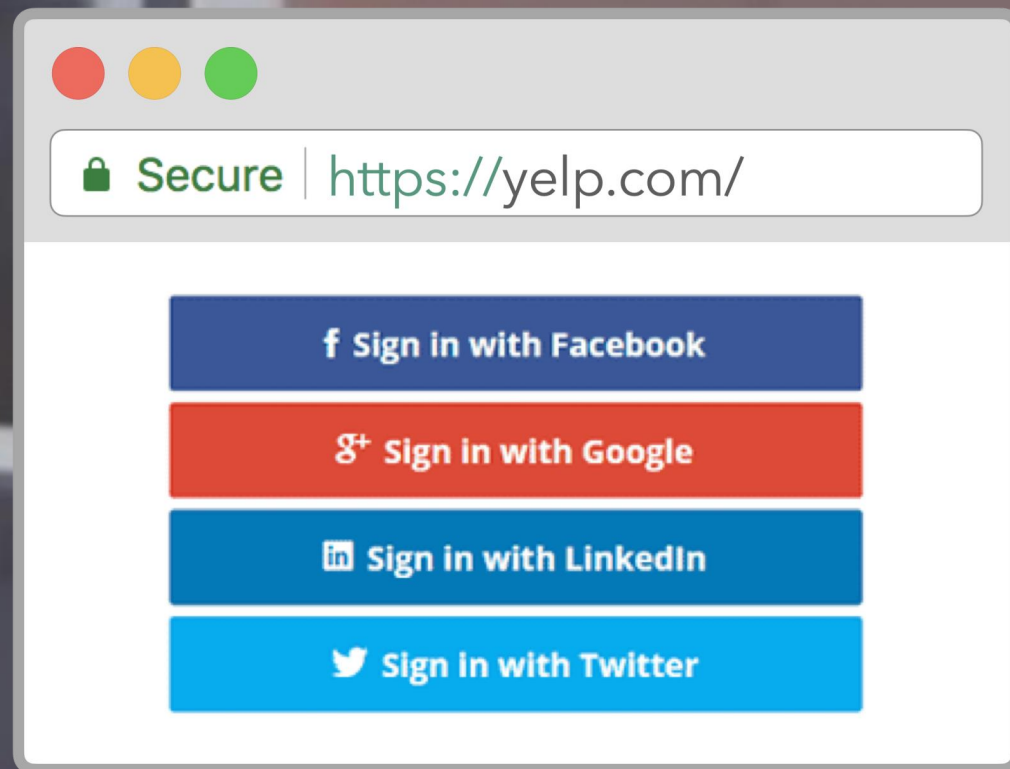
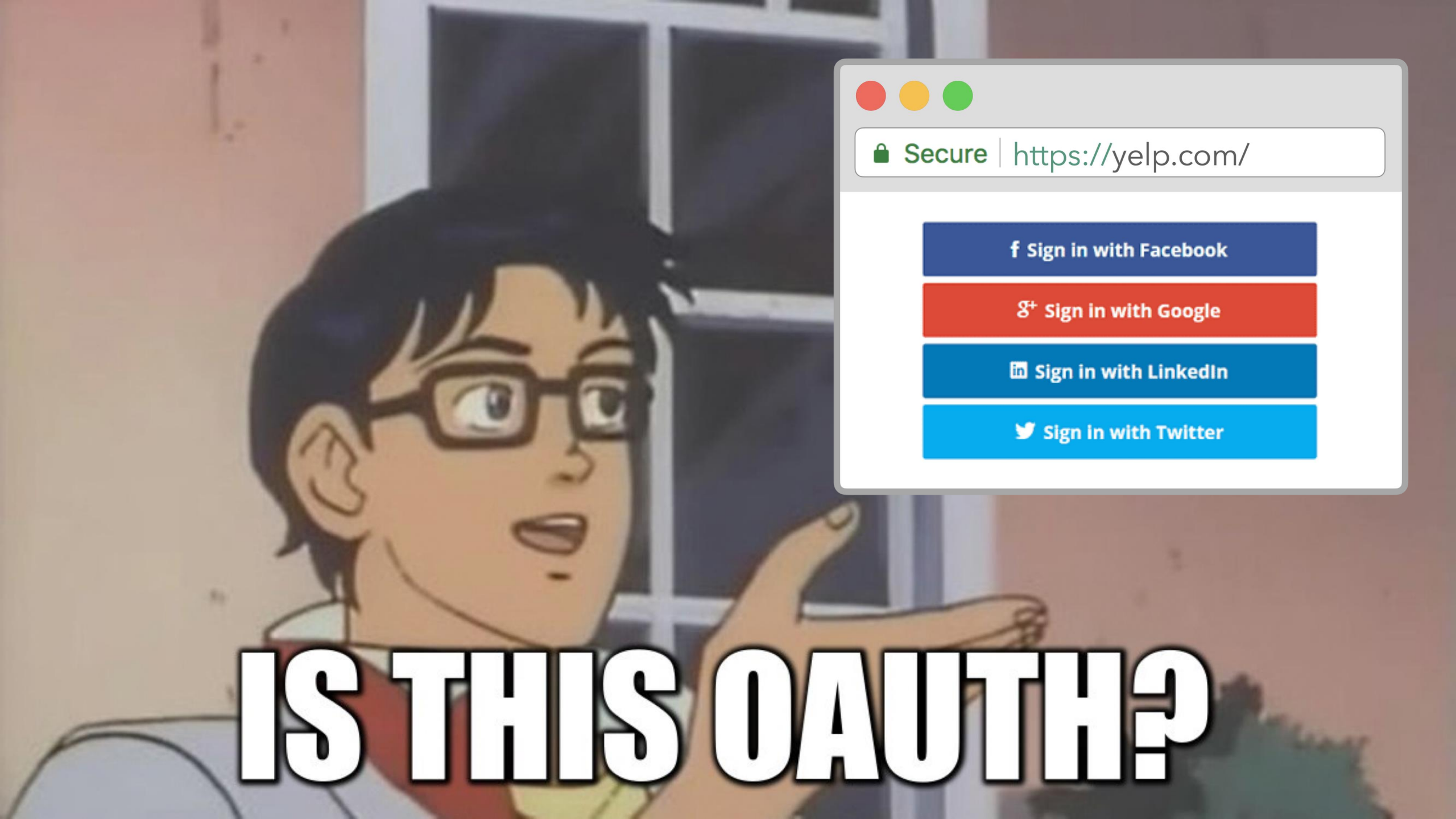
Secure | <https://yelp.com/>

 Sign in with Facebook

 Sign in with Google

 Sign in with LinkedIn

 Sign in with Twitter



IS THIS OAUTH?

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



Your Email Address

(e.g. bob@gmail.com)

Your Gmail Password

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

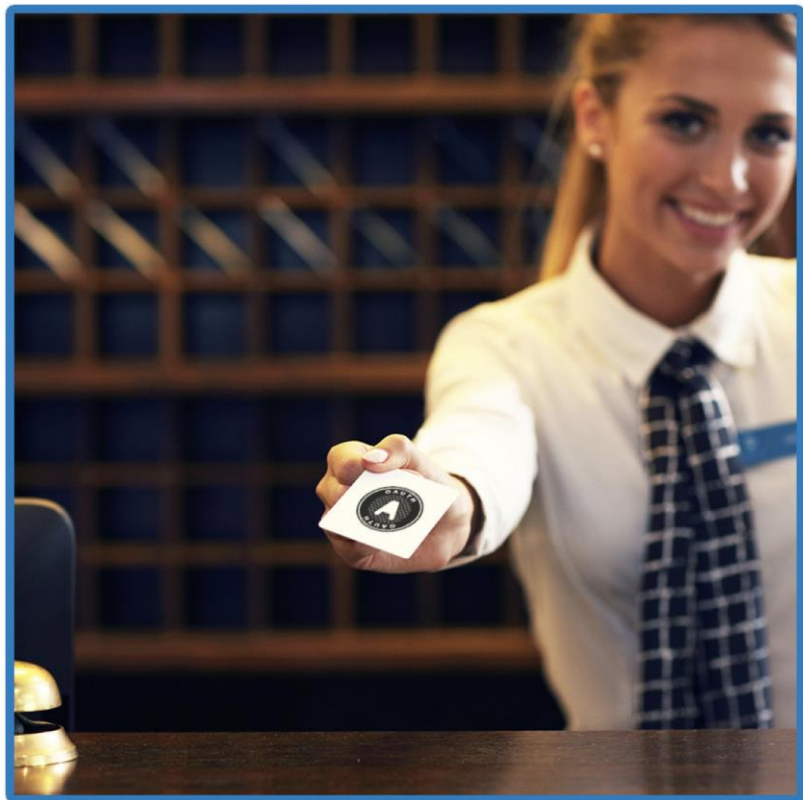
[Skip this step](#)

Check Contacts



Google Contacts

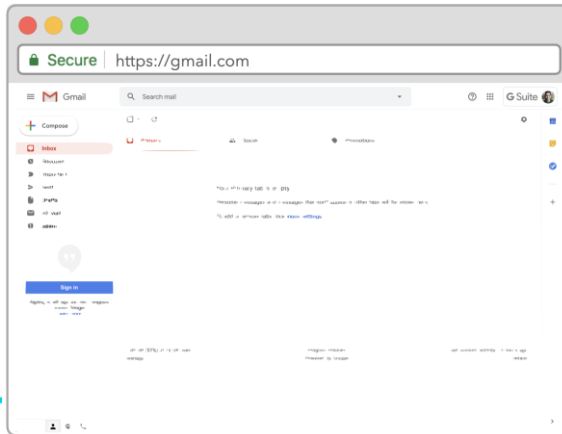
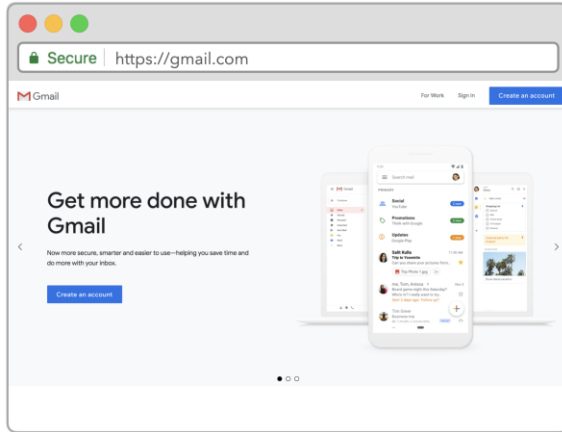




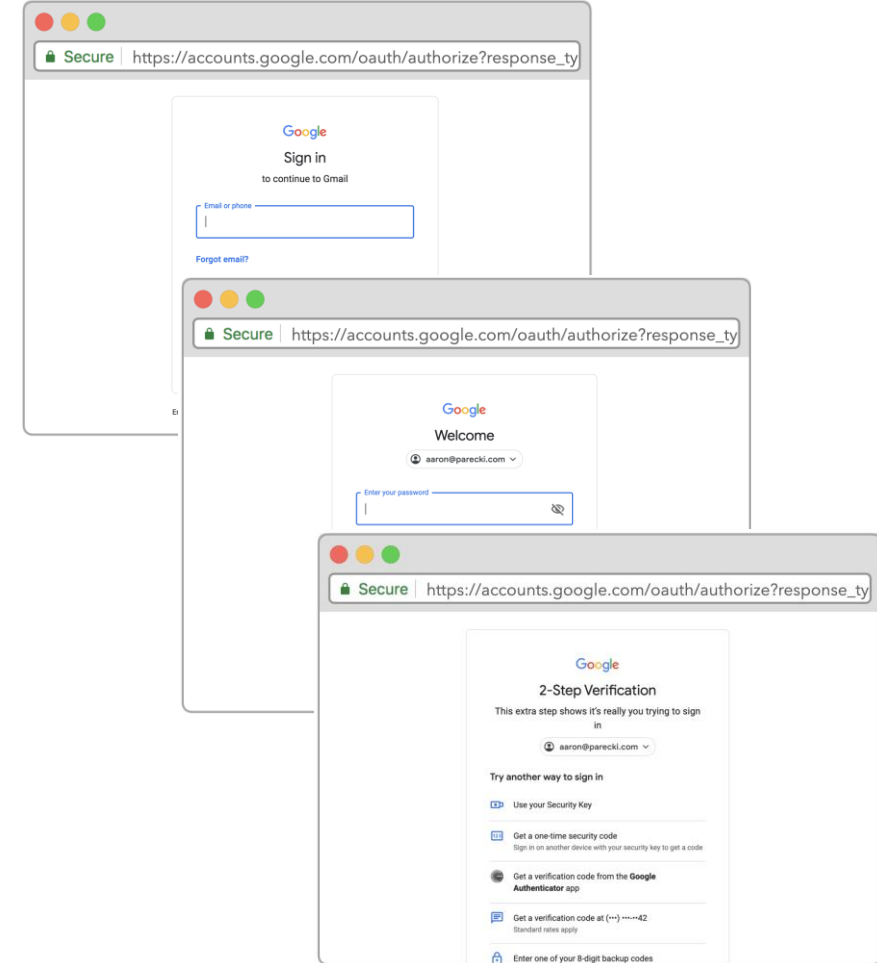


oauth.net/2

Application



OAuth Server







 Sign in with Google

Facebook wants to access your Google Account

 aaron.parecki@gmail.com

This will allow Facebook to:

-  See and download your contacts 
-  See, edit, download, and permanently delete your contacts 

Make sure you trust Facebook

You may be sharing sensitive info with this site or app. You can always see or remove access in your [Google Account](#).

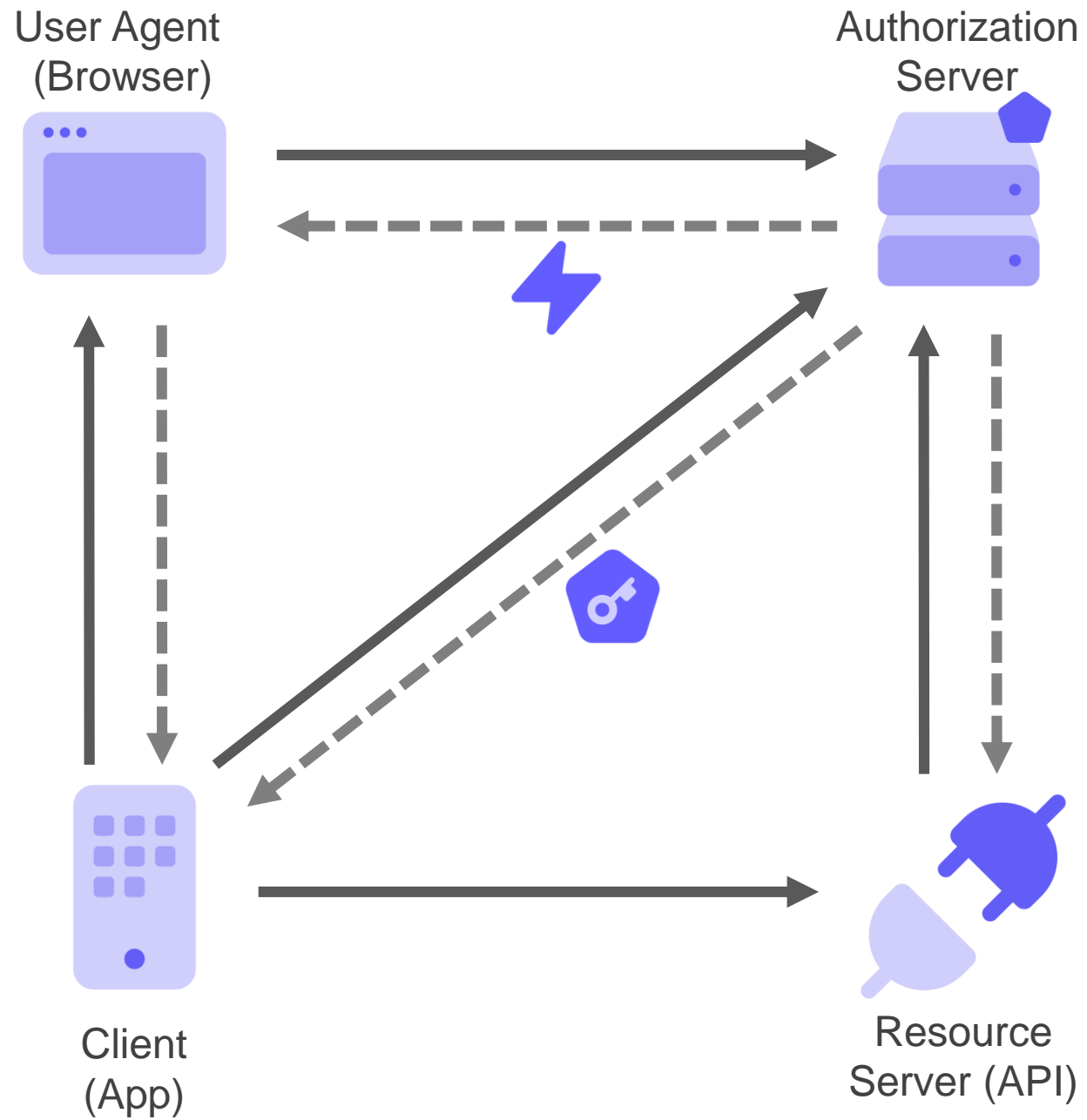
Learn how Google helps you [share data safely](#).

See Facebook's Privacy Policy and Terms of Service.

[Cancel](#)

[Allow](#)

OAuth 2.0



RFC6749 OAuth Core

Authorization Code

Implicit

Password

Client Credentials



RFC6749 OAuth Core

Authorization Code

Implicit

Password

Client Credentials

RFC6750 Bearer Tokens

Tokens in HTTP Header

Tokens in POST Form Body

Tokens in GET Query String

 iPhone



android

RFC6749 OAuth Core

RFC7636

+PKCE

Authorization Code

Implicit

Password

Client Credentials

RFC6750 Bearer Tokens

Tokens in HTTP Header

Tokens in POST Form Body

Tokens in GET Query String

RFC6749 OAuth Core

Authorization Code

Implicit

Password

Client Credentials

RFC7636

+PKCE

RFC8252

PKCE for mobile

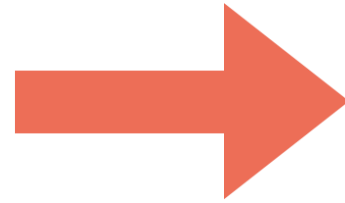
RFC6750 Bearer Tokens

Tokens in HTTP Header

Tokens in POST Form Body

Tokens in GET Query String

RFC6749 OAuth Core



Authorization Code

Implicit

Password

Client Credentials

RFC7636
+PKCE

RFC8252
PKCE for mobile

RFC6750 Bearer Tokens

Tokens in HTTP Header

Tokens in POST Form Body

Tokens in GET Query String



RFC6749 OAuth Core

Authorization Code

~~Implicit~~

~~Password~~

Client Credentials

Security BCP

PKCE for
confidential
clients

RFC7636

+PKCE

Browser App BCP

PKCE for SPAs

RFC8252

PKCE for mobile

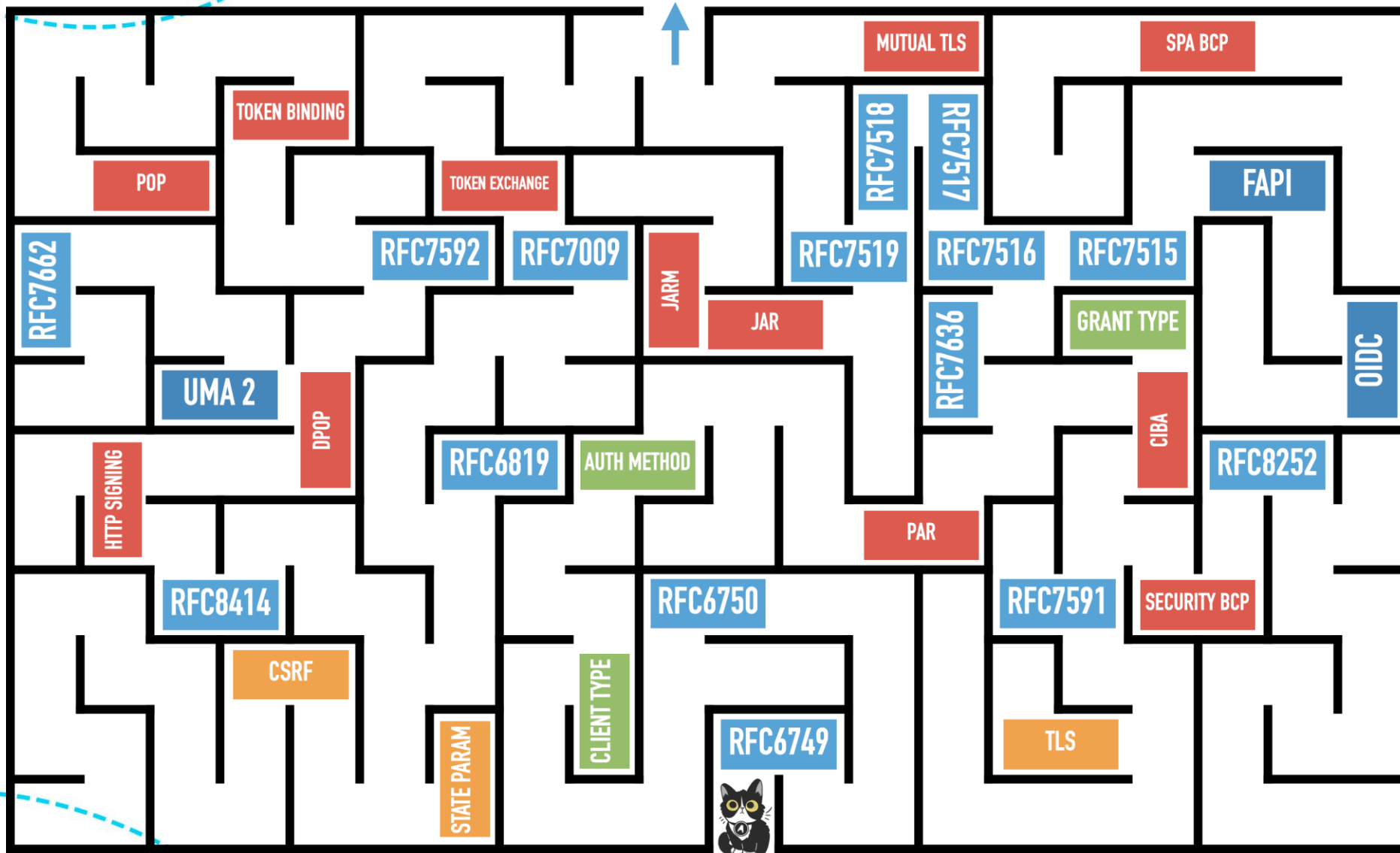
RFC6750 Bearer Tokens

Tokens in HTTP Header

Tokens in POST Form Body

~~Tokens in GET Query String~~

BUILDING YOUR APPLICATION



Authorization Code +PKCE

Client Credentials

Tokens in HTTP Header

Tokens in POST Form Body

OAuth 2.1

oauth.net/2.1



OAuth 2.1 Goals

- Consolidate the OAuth 2.0 specs, adding best practices, removing deprecated features
- Capture current best practices in OAuth 2.0 under a single name
- Add references to extensions that didn't exist when OAuth 2.0 was published

OAuth 2.1 Non-Goals

- No new behavior defined by OAuth 2.1
- Don't include anything experimental, in progress, or not widely implemented

Recent OAuth Extensions



User Agent
(Browser)

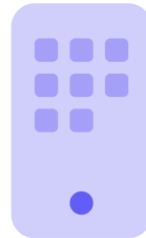
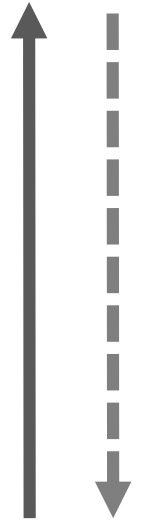


Authorization
Server

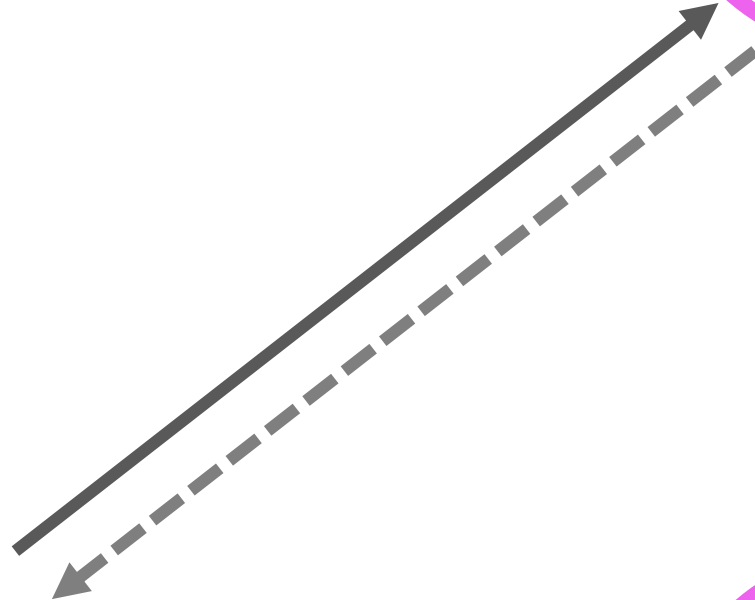


RFC9068

JWT Profile for Access Tokens



Client
(App)



Resource
Server (API)

RFC9068

JWT Profile for Access Tokens

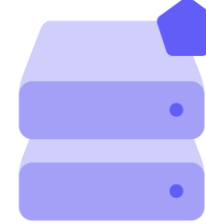
oauth.net/2/jwt-access-tokens

```
{ "typ": "at+JWT", "alg": "RS256", "kid": "RjEw0w0A" }  
{  
  "iss": "https://authorization-server.com/",  
  "sub": "5ba552d67",  
  "aud": "https://rs.example.com/",  
  "exp": 1544645174,  
  "client_id": "s6BhdRkqt3_",  
  "scope": "openid profile email"  
}
```

User Agent
(Browser)

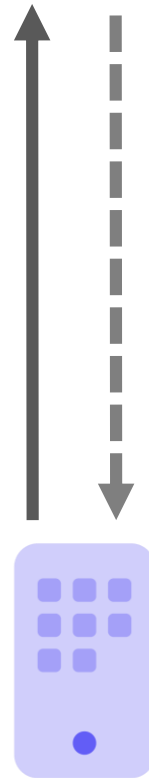


Authorization
Server



RFC9126

**Pushed
Authorization
Requests**



Client
(App)



Resource
Server (API)

Pushed Authorization Requests (PAR)

oauth.net/2/pushed-authorization-requests

- Typically, the authorization request is sent in the front channel
- Front channel is susceptible to inspection and modification
- PAR initiates the OAuth flow from the back channel

Pushed Authorization Requests (PAR)

oauth.net/2/pushed-authorization-requests

Instead of:

```
GET /authorize?response_type=code
&client_id=s6BhdRkqt3&state=af0ifjsldkj
&redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb HTTP/1.1
Host: as.example.com
```

Push the request to the AS:

```
POST /as/par HTTP/1.1
Host: as.example.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic czZCaGRSa3F0Mzo3RmpmcDBaQnIxS3REUmJuZlZkbUl3

response_type=code
&client_id=s6BhdRkqt3&state=af0ifjsldkj
&redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
```

User Agent
(Browser)

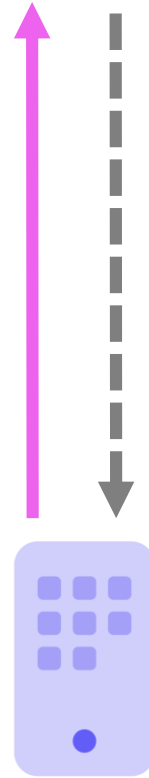


Authorization
Server



RFC9101

**JWT
Authorization
Requests**



Client
(App)



Resource
Server (API)

User Agent
(Browser)

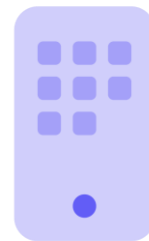
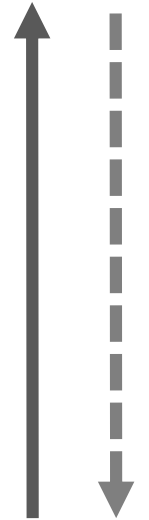


Authorization
Server

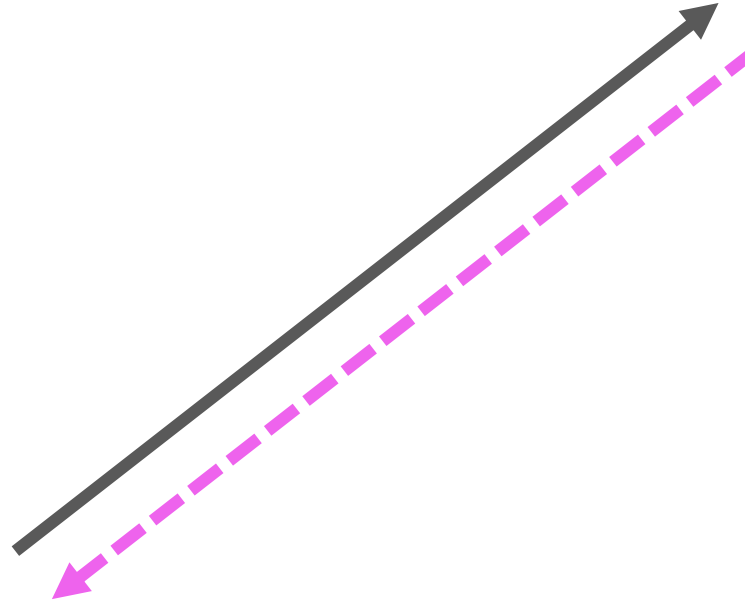


RFC9207

**Authorization
Server
Issuer
Identification**



Client
(App)



Resource
Server (API)

Authorization Server Issuer Identification

datatracker.ietf.org/doc/html/rfc9207

- Clients that interact with multiple authorization servers are at risk of a mixup attack
- Adds a new parameter to the authorization response redirect from the AS to the client
- Simple to implement on both clients and servers

Nearly-Final Specifications



User Agent
(Browser)

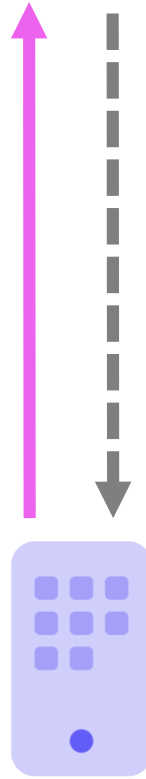


Authorization
Server

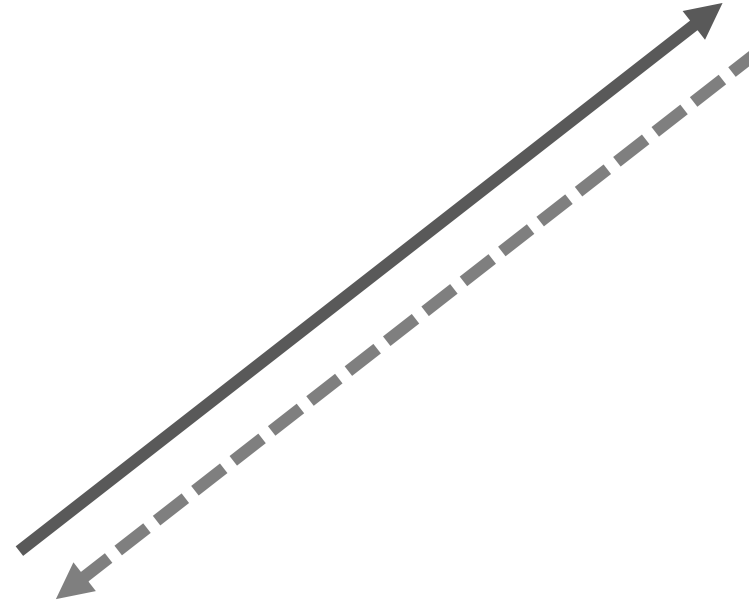
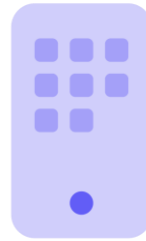


**RFC
Pending**

**Rich
Authorization
Requests**



Client
(App)



Resource
Server (API)

RFC
Pending

Rich Authorization Requests

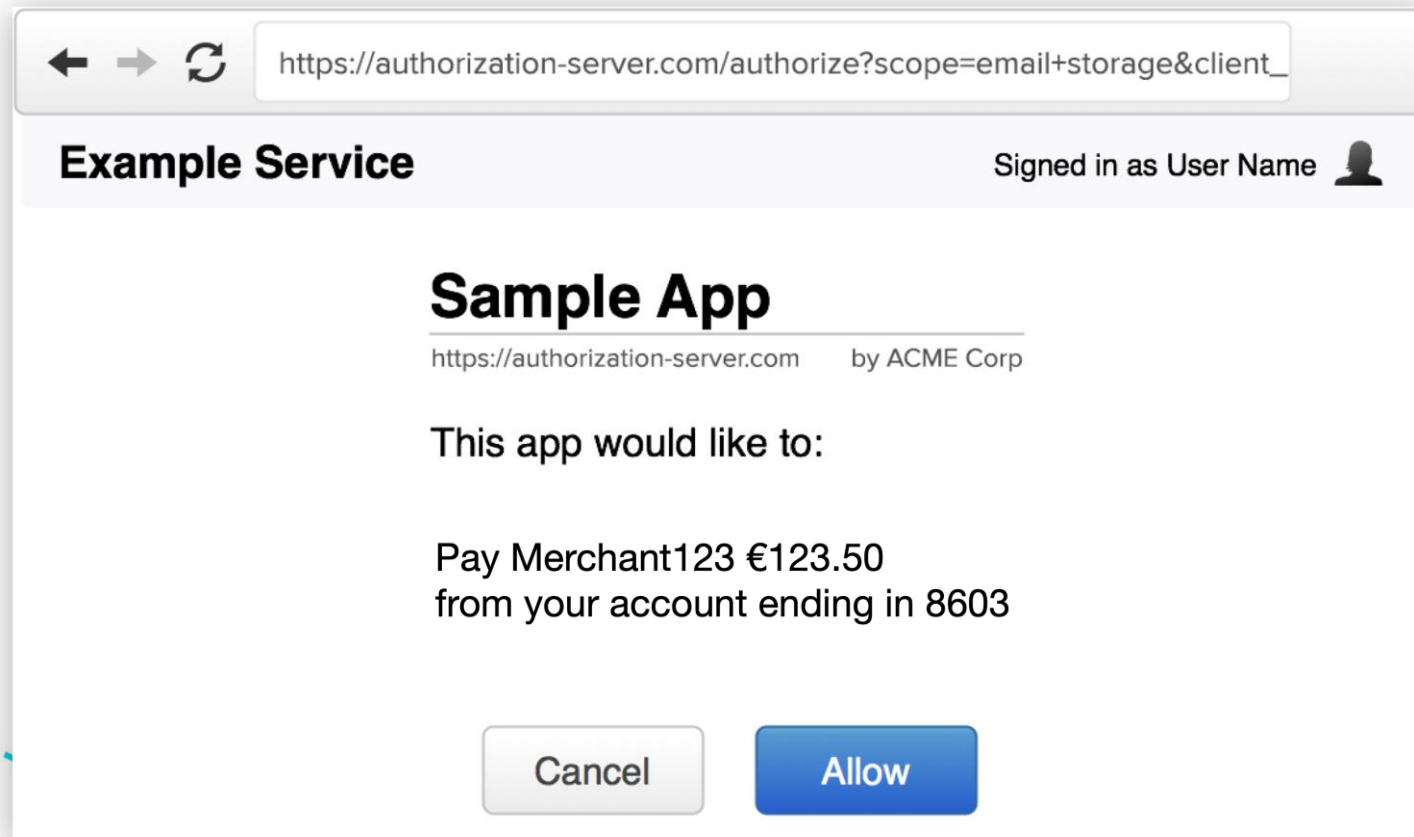
oauth.net/2/rich-authorization-requests

- OAuth “scope” is limited
- Need a way to authorize fine-grained transactions or resources
- And present that to the user in the authorization interface

Rich Authorization Requests


oauth.net/2/rich-authorization-requests

RFC
Pending



A screenshot of a web browser window showing an OAuth authorization dialog. The browser's address bar contains the URL `https://authorization-server.com/authorize?scope=email+storage&client_`. The dialog header shows 'Example Service' on the left and 'Signed in as User Name' with a user icon on the right. The main content area displays 'Sample App' as the application name, with the URL `https://authorization-server.com` and 'by ACME Corp' below it. A message states 'This app would like to:' followed by a payment request: 'Pay Merchant123 €123.50 from your account ending in 8603'. At the bottom, there are two buttons: a grey 'Cancel' button and a blue 'Allow' button.

← → ↻ `https://authorization-server.com/authorize?scope=email+storage&client_`

Example Service Signed in as User Name 

Sample App
`https://authorization-server.com` by ACME Corp

This app would like to:

Pay Merchant123 €123.50
from your account ending in 8603

Cancel Allow

RFC
Pending

Rich Authorization Requests

oauth.net/2/rich-authorization-requests

What's new since 2022?

- Editorial changes based on IETF reviews
- Updated references
- Grammar fixes and clarifications

User Agent
(Browser)



Authorization
Server



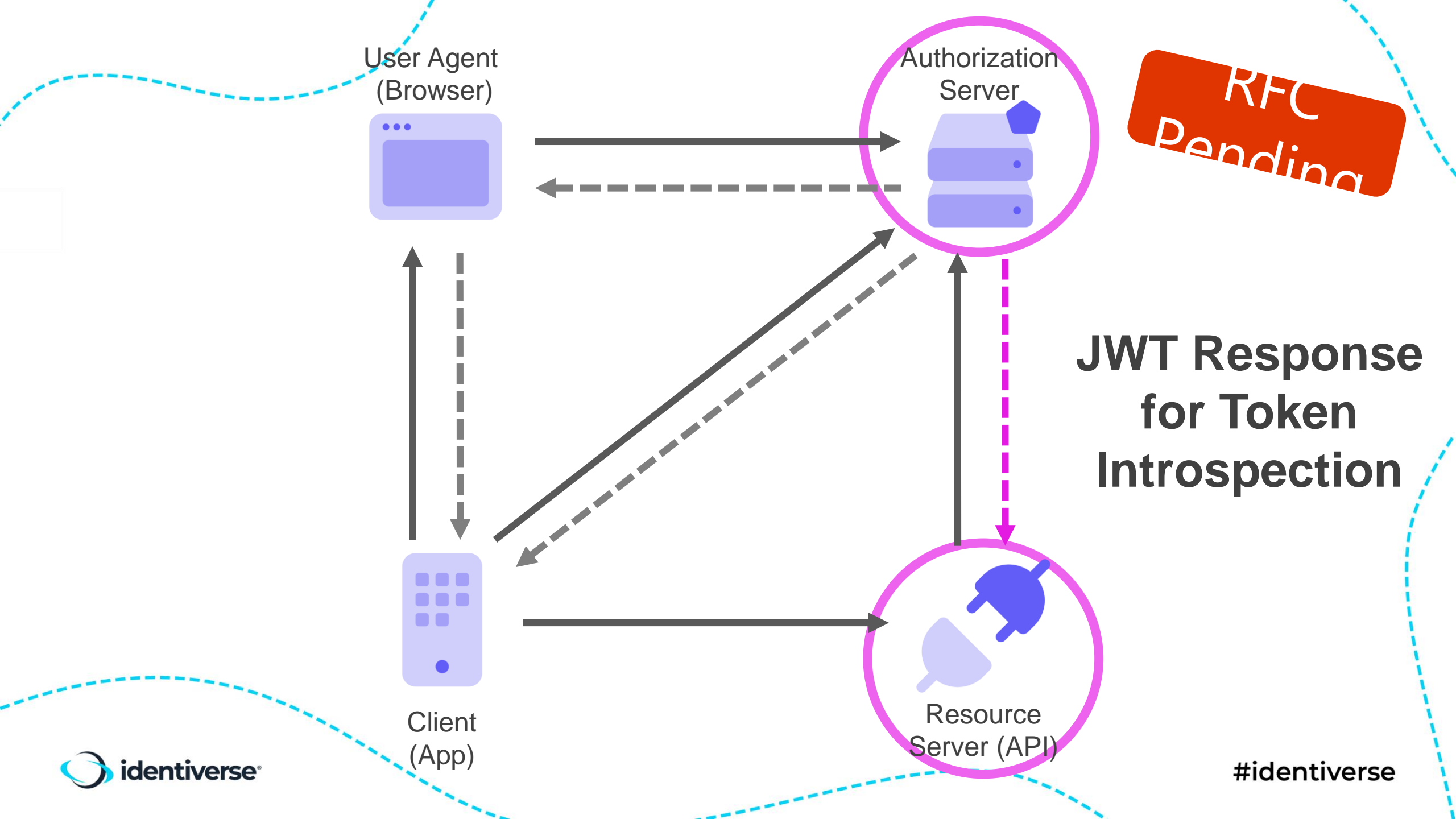
**RFC
Pending**

**JWT Response
for Token
Introspection**



Client
(App)

Resource
Server (API)



RFC
Pending

JWT Response for OAuth Token Introspection

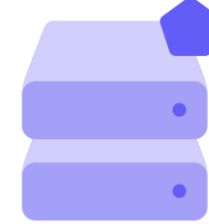
datatracker.ietf.org/doc/html/draft-ietf-oauth-jwt-introspection-response

- Response to a token introspection request is a full signed JWT
- Provides a JWT that can be logged and later used to prove that the AS returned the given introspection response

User Agent
(Browser)

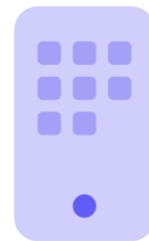
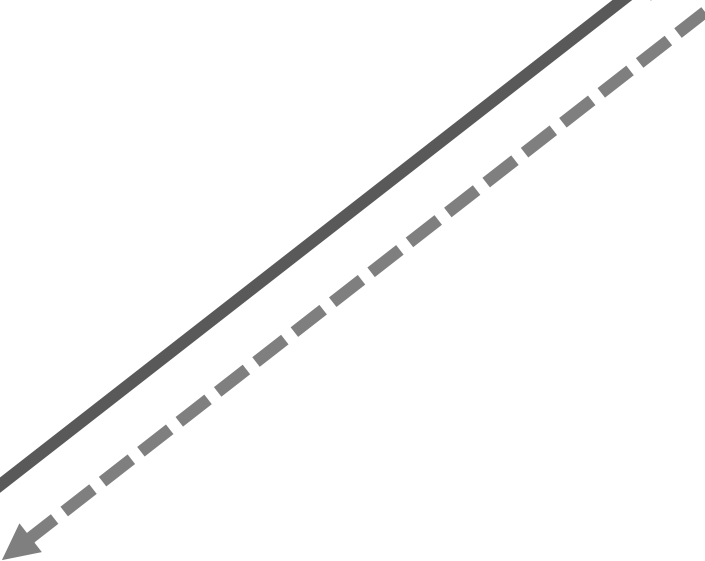
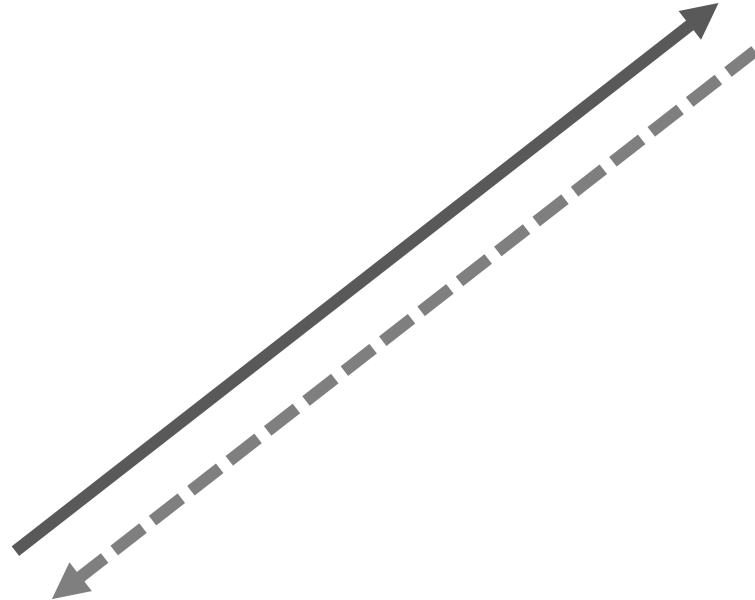
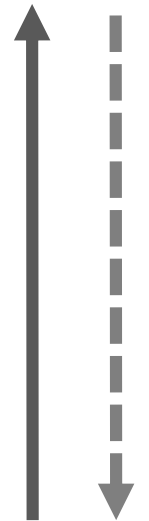


Authorization
Server



**RFC
Pending**

**Step-Up
Authentication
Challenge**



Client
(App)



Resource
Server (API)

RFC
Pending

Step-Up Authentication Challenge

<https://datatracker.ietf.org/doc/html/draft-ietf-oauth-step-up-authn-challenge>

- A Resource Server can respond with an error telling the client to re-authenticate the user or get a higher level authentication
- The client sends the user through the OAuth flow again to get a new access token
- Adopted by the OAuth WG last year, now pending final reviews

Sender-Constrained Access Tokens





LinusTechTipsTemp

@temporaryhandle 15.3M subscribers 14 videos



HOME

VIDEOS

LIVE

PLAYLISTS

COMMUNITY

CHANNELS

ABOUT

2 live streams



OpenAI GPT-4: The Game-Changing AI Technology

LinusTechTipsTemp • 1.1K watching

Unlocking the Power of AI: Everything You Need to Know About OpenAI and ChatGPT - The Revolutionary Chatbot Changing the Game!" In this video, we dive deep into the world of AI with OpenAI...



Elon Musk: Is Bitcoin Back? Bitcoin & Ethereum set to EXPLODE in 2023!

LinusTechTipsTemp • 677 watching

A cryptocurrency, crypto-currency, or crypto is a digital currency designed to work as a medium of exchange through a computer network that is not reliant on any central authority, such as...

YouTube

Search

Use the QR Code or BTC Address to Join:

- 1Musk1YrsC1z3LYNAtqrsLURe7Ub8T2SWn

To participate, you just need to send 0.1 BTC to 20 BTC to the contribution address and we will immediately send you back 0.2 BTC to 40 BTC to the address you sent it from.

If you send 0.1+ BTC, you will receive 0.2+ BTC back
If you send 0.5+ BTC, you will receive 1+ BTC back
If you send 1+ BTC, you will receive 2+ BTC back
If you send 5+ BTC, you will receive 10+ BTC back
If you send 10+ BTC, you will receive 20+ BTC back
If you send 20 BTC, you will receive 40 BTC back

More info on spacex-bitcoin.org

You can only participate once.

#SpaceX #DM2 #Splashdown

Elon Musk Interview from Air Warfare Symposium about SpaceX Crew Dragon & NASA 2020

36,051 watching now

1.1K

175

SHARE

SAVE

...



Live News

18.4K subscribers

JOIN

SUBSCRIBE

Site: <https://cutt.ly/6dRyRiu>





**Token
Request**

POST /token
grant_type=authorization_code
client_id=X
code=Y
redirect_uri=Z

**Token
Response**

```
{  
  "token_type": "Bearer",  
  "access_token": "X8aMHeW922"  
}
```

**Resource
Request**

GET /resource
Authorization: Bearer X8aMHeW922

**Token
Request**

POST /token
grant_type=authorization_code
client_id=X
code=Y
redirect_uri=Z

**Token
Response**

```
{  
  "token_type": "Bearer",  
  "access_token": "X8aMHeW922"  
}
```

**Resource
Request**

GET /resource
Authorization: Bearer X8aMHeW922

Sender-Constrained Access Tokens

Problem: Bearer tokens can be used by anyone who can steal one

Solution: Require some sort of authentication of the client instance in order to use an access token

Proof of Possession

- **Mutual TLS** – RFC 8705
 - Client is identified by the fingerprint of the public key
- **DPoP** – *OAuth WG* - Near publication
 - Bunds an OAuth access token to a private key
- **Signed HTTP Requests** – *HTTP WG* - Near publication
 - Choose what parts of an HTTP message to sign
- **OAuth Token Binding** - Expired Draft
 - TLS layer, collecting existing TLS RFCs
 - No activity since October 2018
- **JWT Pop Tokens** - Expired Draft
 - Another HTTP header with signed JWS

DPoP

oauth.net/2/dpop

- Binds the access token to a private key used by the client
- Specifically for OAuth 2.0, not a general-purpose signing mechanism

DPoP

oauth.net/2/dpop

POST /token

Host: authorization-server.com

DPoP: eyJ0eXAiOiJ...

grant_type=authorization_code

&client_id=X

&code=Y

&code_verifier=Z

DPoP

oauth.net/2/dpop

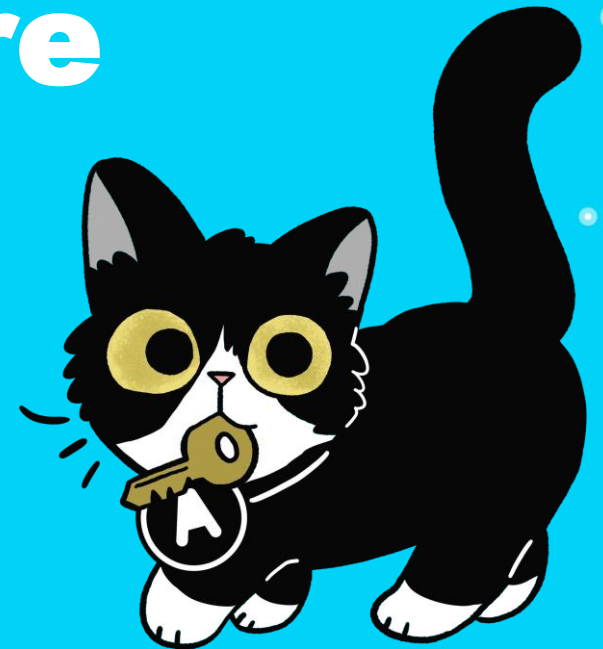
```
GET /api
```

```
Host: example.com
```

```
Authorization: DPoP {ACCESS_TOKEN}
```

```
DPoP: eyJ0eXAiOiJ...
```

Selective Disclosure



Selective Disclosure JWT

datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/

- Currently, JWTs are either entirely encrypted, or only signed

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}  
.  
{  
  "sub": "1234567890",  
  "name": "Aaron Parecki",  
  "iat": 1516239022  
}  
.  
SIGNATURE
```

Selective Disclosure JWT

datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/

- Currently, JWTs are either entirely encrypted, or only signed
- SD-JWT enables issuing a JWT whose values are obscured until intentionally released by the “holder” (subject, owner)
- Each claim in a JWT can be disclosed individually without disclosing the other claims

Selective Disclosure JWT

datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/

- Proposed in June 2022
- Has already been iterated on significantly to improve privacy-preserving characteristics

Selective Disclosure JWT

datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/

```
{  
  ...  
  "name": "Aaron Parecki"  
  ...  
}
```



```
["(SALT)", "name", "Aaron Parecki"]
```

Selective Disclosure JWT

datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/

```
["(SALT)", "name", "Aaron Parecki"]
```



Base64

```
WyJiZTJlOTZhMmFjOWU3NWl3NmNjOGEiLCJuYW1lIiwiaWF0Ij0iYb24gUGFyZWNaSjd
```

The diagram illustrates the process of selectively disclosing a JWT token. It starts with a URL to a draft document: `datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/`. Below this is a long JWT token: `WyJiZTJlOTZhMmFjOWU3NWl3NmNjOGEiLCJuYW1lIiwiQWFiYb24gUGFyZWNaSjd`. A large blue arrow points down from the token to its SHA256 hash: `6PLkXNaDRUdnmleO7lpEd42DvsYYOzPdvLf09J8QR78`. The text "SHA256" is placed next to the arrow. The entire diagram is framed by a dashed blue border.

Selective Disclosure JWT

`datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/`

`WyJiZTJlOTZhMmFjOWU3NWl3NmNjOGEiLCJuYW1lIiwiQWFiYb24gUGFyZWNaSjd`

↓ SHA256

`6PLkXNaDRUdnmleO7lpEd42DvsYYOzPdvLf09J8QR78`

datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/

WyJiZTJlOTZhMmFjOWU3NWl3NmNjOGElLCJuYW1lIiwiaWF0eSIsImVudCI6Im91dG8gaW5kaXQyZWYyZW5raSId



SHA256

6PLkXNaDRUdnmlE07lpEd42DvsYYOzPdvLf09J8QR78

Selective Disclosure JWT

datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/

6PLkXNaDRUdnmleO7lpEd42DvsYYOzPdvLf09J8QR78



```
{
  "_sd": [
    "6PLkXNaDRUdnmleO7lpEd42DvsYYOzPdvLf09J8QR78"
  ],
  "iss": "https://example.com/issuer",
  "iat": 1684383103,
  "exp": 1684469503,
  "sub": "8Fz3g9LZhVV0DH7njUnKB3eVwan4HOu6ht7c83fK6wQ",
  "_sd_alg": "sha-256"
}
```

References,
upcoming specifications
and events:

oauth.net

THANK YOU!

