# FAPI 2.0

Torsten Lodderstedt/Daniel Fett, yes.com

# Objective

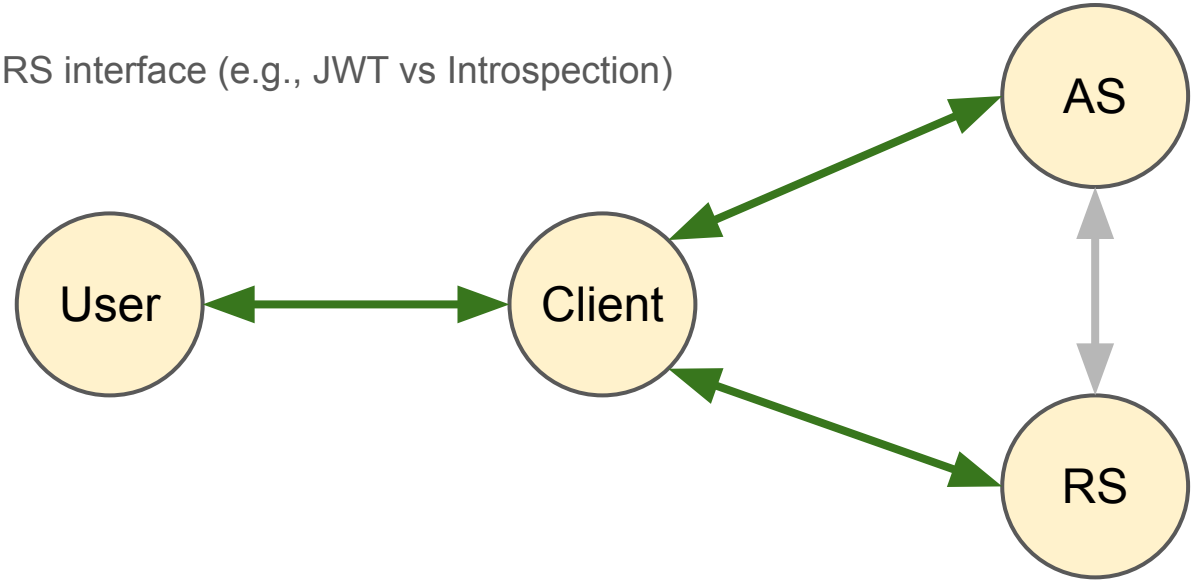
Develop an interoperable security protocol for authorization of access to security- & privacy-critical APIs, e.g., financial or health services

# Requirements

- Based on well-defined threat model
- Compliant with OAuth Security Best Current Practice
- User & Developer friendly
- Support for fine-grained & transactional consent
- Consent lifecycle management
- Conformance can be tested automatically
- Versatility re communication channels (on device, POS, …)
- Leverages international standards

# Scope

- In scope:
  - Interoperability for client to AS interface
  - Security mechanisms between client and RS
- Out of scope:
  - Interoperability for AS to RS interface (e.g., JWT vs Introspection)



# Security Goals

## Primary Goals

**Authentication**
No attacker is able to login at a client under the identity of a user.

**Authorization**
No attacker can access resources belonging to a user.

**Session Integrity**
No attacker is able to force a user to use resources of or identify as the attacker.

## Non-Repudiation Goals

“the assurance that someone cannot deny something”

- Authorization Requests
- Authorization Responses
- ID Token Contents
- Introspection Responses
- Userinfo Responses
- Resource Requests
- Resource Responses

# Threat Model

## Attacker Capabilities

**Network Attacker**
Has full control over the network.

+ **Read secrets from URLs**

+ **Read token and resource requests/responses**

+ **Tamper with Resource Responses**

But not: Breaking TLS or distribution false keys.

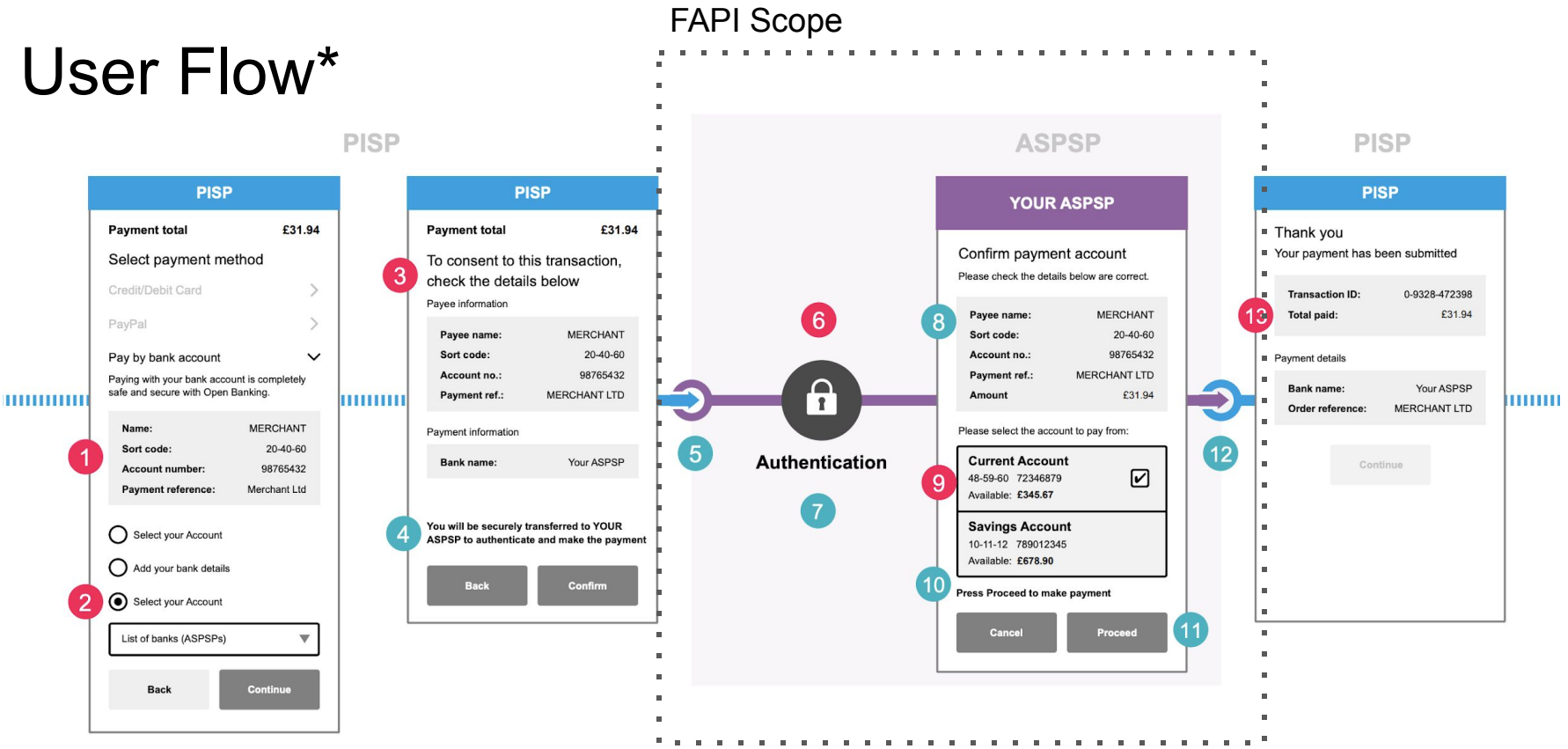
## Evaluation

- Clear definition of security goals and attacker capabilities
- Prerequisite for formal evaluation of security
- Together with researchers, based on previous research
- Goal: Proof of security of FAPI Evolution

# FAPI Components

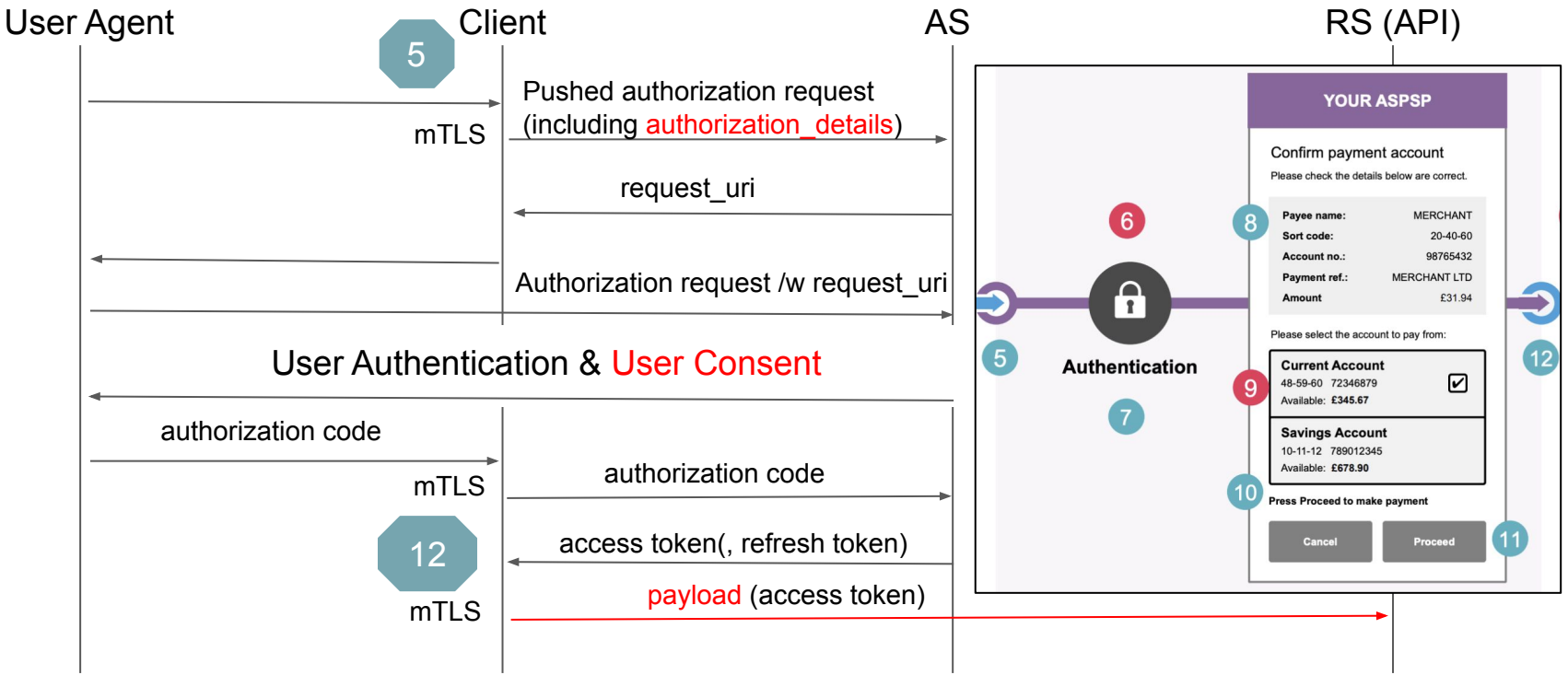
- Baseline Profile
  - Redirect based authorization flow for all kinds of apps (x2web and x2app)
  - RFC 6749 & RFC 6750, RFC 7636 (PKCE), JAR, PAR, RAR, RFC 7009, RFC 8705 (mTLS)
  - Implementation advice
    - app2app redirect
    - on authorization grant (aka consent) lifecycle management and dynamic linking
- Advanced Profile
  - JARM, OpenId Connect (ID Token as detached signature)
  - Message Signing
- CIBA (decoupled authorization flow)
- Grant Management protocol & API

# User Flow*




*Source: https://www.openbanking.org.uk/wp-content/uploads/Customer-Experience-Guidelines.pdf

# Generic Baseline Profile Flow




API/Ecosystem specific

# Pushed Authorization Request (Example)

```
POST /as/par HTTP/1.1
Host: as.example_aspsp.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
&code_challenge_method=S256
&code_challenge=K2-ltc83acc4h0c9w6ESC_rEMTJ3bww-uCHaoeK1t8U
&authorization_details=%5B%7B%22type%22%3A%22payment%5Finitiation%22%2C%22locations%22%3A%5B%
22https%3A%2F%2Fexample%5Faspsp%2Ecom%2Fpayments%22%5D%2C%22instructedAmount%22%3A%7B%
22currency%22%3A%22GBP%22%2C%22amount%22%3A%2231%2E94%22%7D%2C%22creditorName%22%3A%22M
erchant%22%2C%22creditorAccount%22%3A%7B%22no%22%3A%2298765432%22%7D%2C%22remittanceInformation
Unstructured%22%3A%22MERCHANT%20LTD%22%7D%5D
```

# authorization_details (Example)


```
[
   {
      "type":"payment_initiation",
      "locations":[
         "https://api.example_aspsp.com/payments"
      ],
      "instructedAmount":{
         "currency":"GBP",
         "amount":"31.94"
      },
      "creditorName":"Merchant",
      "creditorAccount":{
         "no":"98765432"
      },
      "remittanceInformationUnstructured":"MERCHANT LTD"
   }
]
```

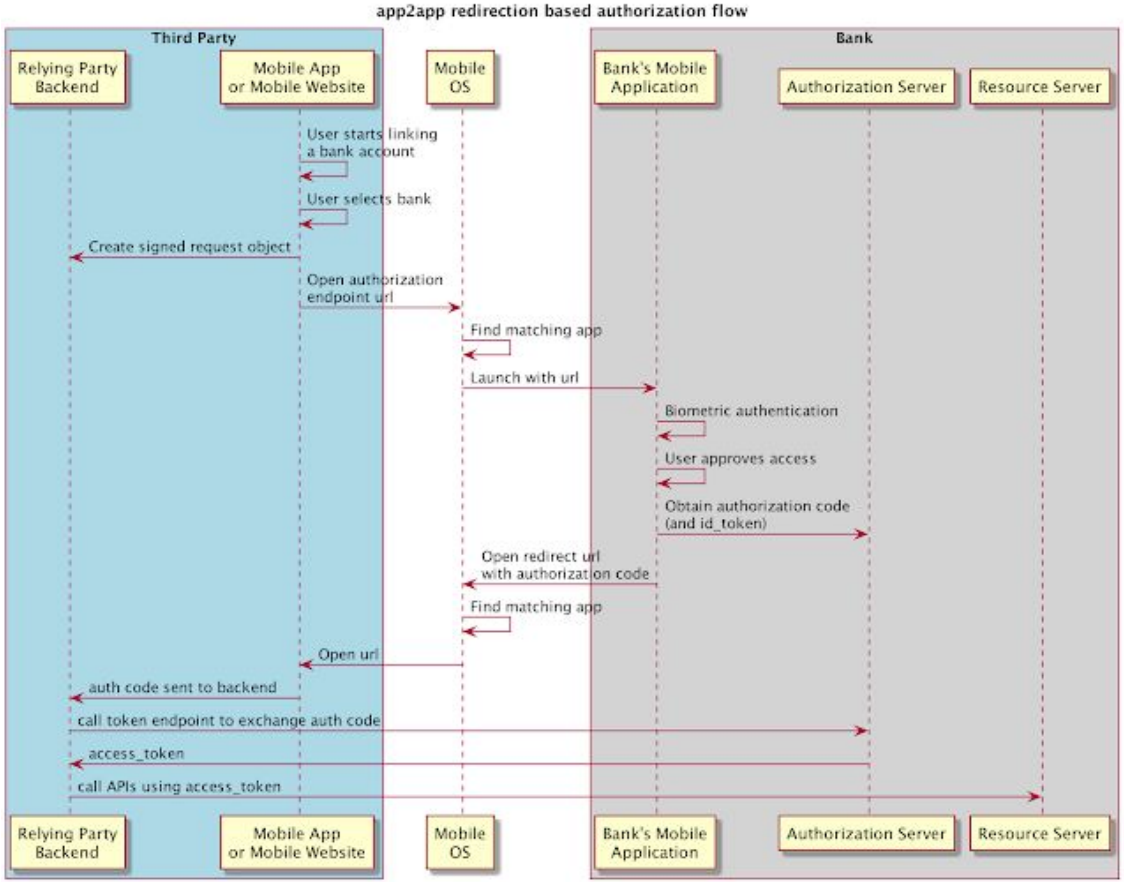
# Dynamic Linking

- AS adds authorization details to access token
  (or token introspection response)
- including user selected data
  (e.g. account)
- RS enforces dynamic linking


```
{
   "iss":"https://as.example_aspsp.com",
   "sub":"24400320",
   "aud":"a7AfcPcsl2",
   "exp":1311281970,
   "acr":"psd2_sca",
   "txn":"8b4729cc-32e4-4370-8cf0-5796154d1296",
   "authorization_details":[
      {
         "type":"payment_initiation",
         "locations":[
            "https://api.example_aspsp.com/payments"
         ],
         "instructedAmount":{
            "currency":"GBP",
            "amount":"31.94"
         },
         "creditorName":"Merchant",
         "creditorAccount":{
            "no":"98765432"
         },
         "remittanceInformationUnstructured":"MERCHANT LTD"
      }
   ],
   "debtorAccount":{
      "no":"48-59-60 72346879",
      "user_role":"owner"
   }
}
```

# App 2 App Redirect

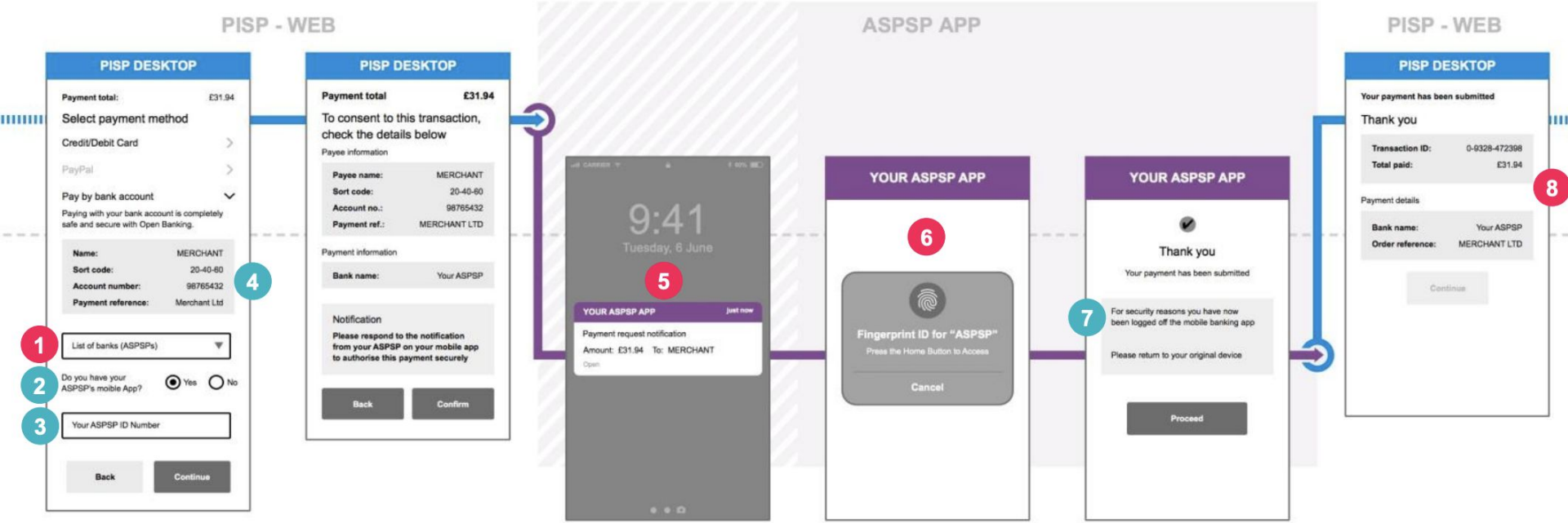
- Standard OAuth/OIDC using universal/app links will automatically open AS app if available
- Otherwise, standard web flow is used




Source: https://fapi.openid.net/2019/10/21/implementing-app-to-app-authorisation-in-oauth2-openid-connect/

# POS, Kiosk, Call Center Use Cases (CIBA)

- Including setup via web flow and subsequent transactions via CIBA

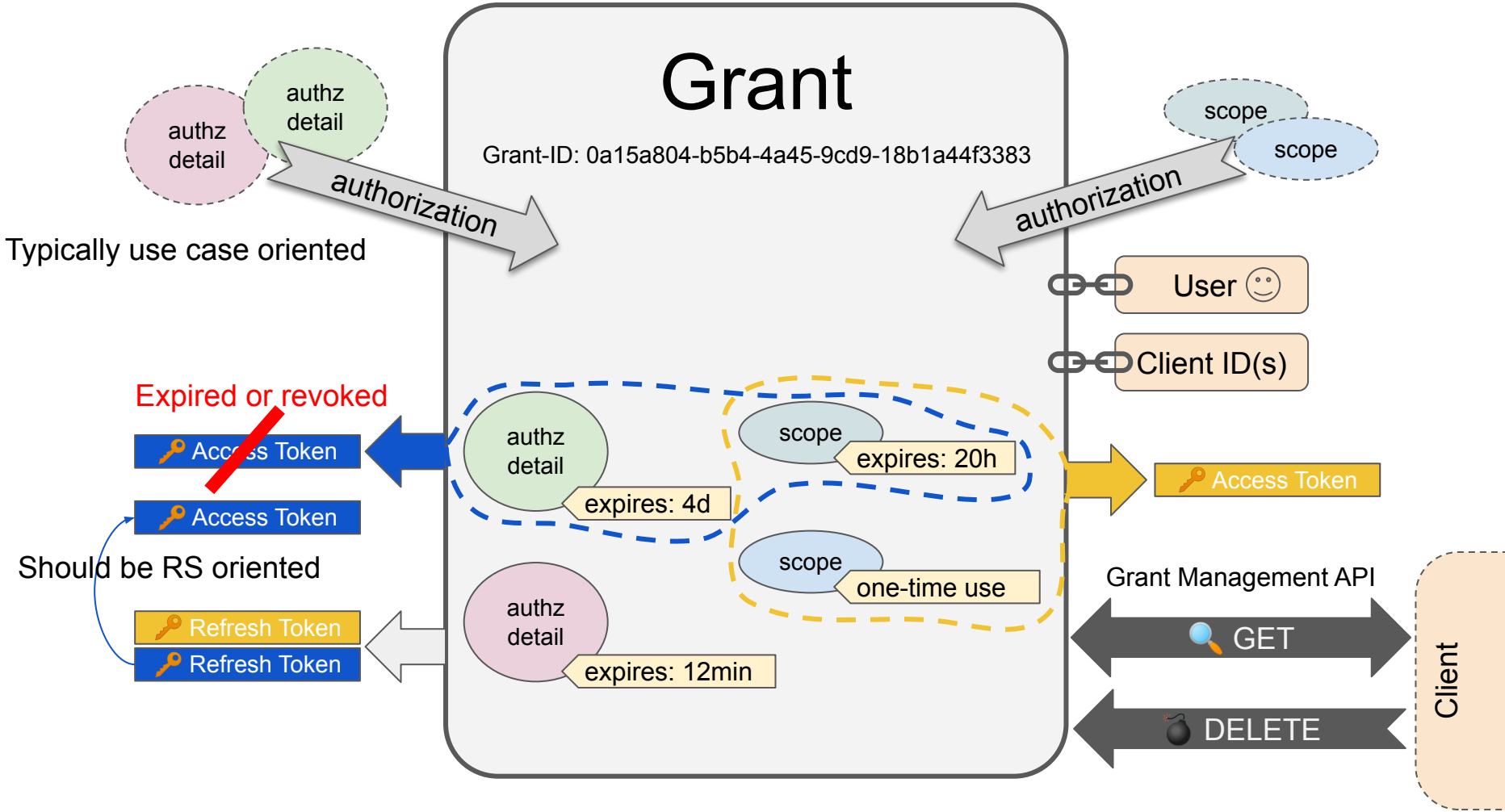



*Source: https://www.openbanking.org.uk/wp-content/uploads/Customer-Experience-Guidelines.pdf

# Grant Management

# Grant Management

- Grant: the set of permissions confirmed by the owner of services or data for a certain client
- Objectives
  - Make grant (status) accessible and manageable by clients
  - Support concurrent, independent grants
- Proposal
  - Define OAuth extension to make **grants** (including all authorization details) identifiable and manageable
  - Allow client to use independent grants for the same user


Grant
Grant-ID: 0a15a804-b5b4-4a45-9cd9-18b1a44f3383
authz detail
authz detail
authorization
Typically use case oriented
scope
scope
authorization
User
Client ID(s)
Expired or revoked
Access Token
Access Token
Should be RS oriented
Refresh Token
Refresh Token
authz detail
expires: 4d
scope
expires: 20h
scope
one-time use
authz detail
expires: 12min
Access Token
Grant Management API
GET
DELETE
Client

# Grant Management (request grant id)

(Pushed) Authorization Request)

```
POST /as/par HTTP/1.1
    Host: as.example.com
    Content-Type: application/x-www-form-urlencoded
    Authorization: Basic czZCaGRSa3F0Mzo3Rm...

    response_type=code&
    client_id=s6BhdRkqt3
    &request_grant_id=default
    &state=af0ifjsldkj
    &redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
    &code_challenge_method=S256
    &code_challenge=K2-ltc83acc4h...
    &authorization_details=%5B%7B%2...
```

Token Response

```
HTTP/1.1 200 OK
  Content-Type: application/json
  Cache-Control: no-cache, no-store

  {
    "access_token": "2YotnFZFEjr1zCsicMWpAA",
    "token_type": "example",
    "expires_in": 3600,
    "refresh_token": "tGzv3JOkF0XG5Qx2TlKWIA",
    “grant_id”:”0a15a804-b5b4-4a45-9cd9-18b1a44f3383”,
    "authorization_details": [...
    ]
  }
```

# Grant Management (API)

## Query

```
GET /grants/0a15a804-b5b4-4a45-9cd9-18b1a44f3383
Host: as.example-bank.com
Authorization: Bearer 2YotnFZFEjr1zCsicMWpAA
```

```
HTTP/1.1 200 OK
Cache-Control: no-cache, no-store
Content-Type: application/json

{
  "authorization_details":[...]
}
```

## Revoke

```
DELETE /grants/0a15a804-b5b4-4a45-9cd9-18b1a44f3383
Host: as.example-bank.com
Authorization: Bearer 2YotnFZFEjr1zCsicMWpAA
```

```
HTTP/1.1 204 No Content
```

# Grant Management (request use of certain grant)

(Pushed) Authorization Request)

```
POST /as/par HTTP/1.1
    Host: as.example.com
    Content-Type: application/x-www-form-urlencoded
    Authorization: Basic czZCaGRSa3F0Mzo3Rm...

    response_type=code&
    client_id=s6BhdRkqt3
    &grant_id=0a15a804-b5b4-4a45-9cd9-18b1a44f3383
    &state=af0ifjsldkj
    &redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
    &code_challenge_method=S256
    &code_challenge=K2-ltc83acc4h...
    &authorization_details=%5B%7B%2...
```

Use cases

- Renew consent (because it is about to be expire)
- Update existing consent
- Ensure authorization process is performed with same user
- Allows identification of user (alternative login hint for CIBA)

# Grant Management (request new/concurrent grant)

(Pushed) Authorization Request)

```
POST /as/par HTTP/1.1
    Host: as.example.com
    Content-Type: application/x-www-form-urlencoded
    Authorization: Basic czZCaGRSa3F0Mzo3Rm...

    response_type=code&
    client_id=s6BhdRkqt3
    &request_grant_id=new
    &state=af0ifjsldkj
    &redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
    &code_challenge_method=S256
    &code_challenge=K2-ltc83acc4h...
    &authorization_details=%5B%7B%2...
```

Description

- Establishes new grant for dedicated use case at client
- Dedicates tokens will be used for this grant (see conceptual model)
- Preserves existing grants with the same client/user combination