A large teal graphic element consisting of a diagonal line that splits the page into a white upper-left section and a teal lower-right section.





DECOUPLED

FLOWS IN

OAUTH 2.0


HELLO!

A bit about me

I'm the CTO at [moneyhub](#) , I'm an active contributor to the  OpenID® FAPI specs. I'm the FAPI WG Liaison Officer to UK . I'm the technical rep for .

 @davidgtonge

 davidgtonge

 dgtonge jwt.davetonge.co.uk

WHY DECOUPLED

Use cases:

- ▶ granting authorisation to remote call centre agent
- ▶ using the strongly authenticated session on a smart device to grant authorisation to another device, e.g. input constrained, or doesn't belong to user or simply doesn't have a strongly authenticated session
- ▶ payments

WHY DECOUPLED

Legislation:

PSD2 and the RTS are worded in such a way as to force banks to provide more than just redirect based flows

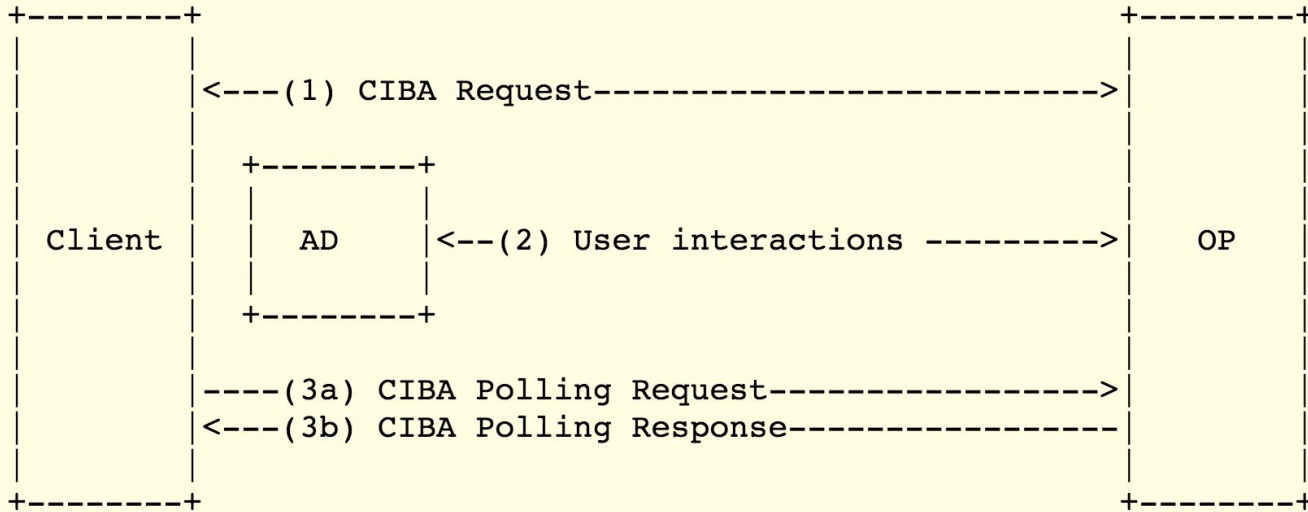
THE PROBLEM

SESSION BINDING

Three flows examined:

- ▶ Client Initiated **Backchannel Authentication**
- ▶ OAuth 2.0 Device Flow
- ▶ Vanilla OAuth 2.0

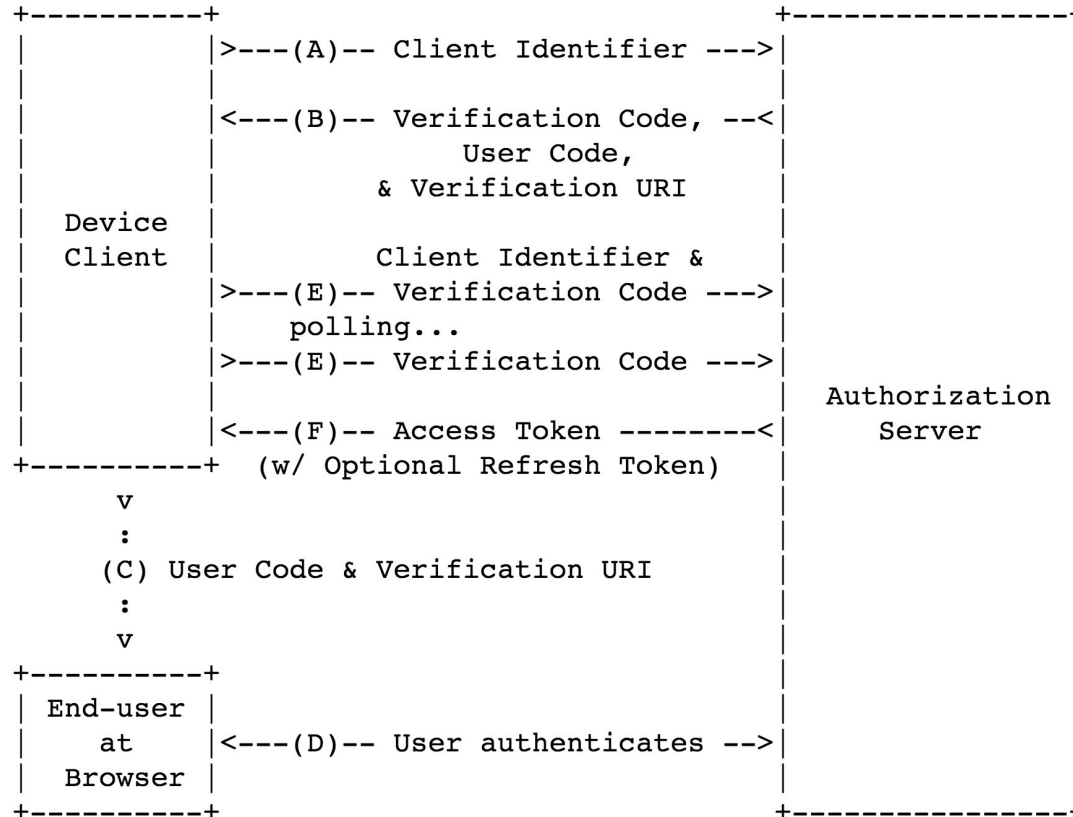
THE CIBA FLOW



THE CIBA FLOW

- ▶ RP sends Backchannel Authentication Request to `/bc_authorize`, with:
 - ▷ `login_hint`
 - ▷ `binding_message` (optional)
 - ▷ `scope`
- ▶ OP responds with Backchannel Authentication Response containing:
 - ▷ `auth_req_id`
- ▶ OP obtains end-user consent/authorization. This process will normally start with a push notification or similar. If a binding message was sent, the OP must show the end-user the binding message and the user must confirm that it is the same as the binding message displayed on the consumption device.
- ▶ RP polls `/token` endpoint with:
 - ▷ `grant_type`:
`"urn:openid:params:modrna:grant-type:backchannel_request"`
 - ▷ `auth_req_id`
- ▶ OP responds with tokens

THE DEVICE FLOW



THE DEVICE FLOW

- ▶ RP sends Device Authorization Request to `/device_authorization`, with
 - ▷ `scope`
- ▶ OP responds with Device Authorization Response containing:
 - ▷ `device_code`
 - ▷ `user_code`
 - ▷ `verification_uri`
- ▶ User navigates to verification uri on their authentication device, authenticates, enters the `user_code` and grants consent.
- ▶ RP polls `/token` endpoint with:
 - ▷ `grant_type: "urn:ietf:params:oauth:grant-type:device_code"`
 - ▷ `device_code`
- ▶ OP responds with tokens

THE DEVICE FLOW

SESSION BINDING CHARACTERISTICS

- ▶ *Same consumption device, different user.* This attack is possible and is called out in the security considerations of the device flow spec. A malicious user just needs to see the user_code displayed on the consumption device and they can hijack the session. For many scenarios this is an acceptable risk that can be mitigated in other ways, for example, showing an informative error message when the real user tries to authenticate.
- ▶ *Different consumption device, same user.* This attack is not possible in the device flow as there is a hidden device_code that is part of the protocol and this is bound to the user code. There are social engineering attacks possible however and these are called out in the security considerations as “remote phishing”. Because of the possibility of such attacks the specification recommends that the user enters the user_code rather than having it prefilled.

THE CIBA FLOW

SESSION BINDING CHARACTERISTICS

- ▶ *Same consumption device, different user.* This attack is difficult in CIBA itself as it is the OP's responsibility to initiate the authentication session with the user. Depending on the type of `login_hint`, this could be possible, but it is outside of the scope of the CIBA spec.
- ▶ *Different consumption device, same user.* In CIBA there is an optional binding_message. Without this it would be possible for a malicious actor to trick the user into authorising access to the wrong consumption device.

CIBA DEEP DIVE

- ▶ Identity
- ▶ Session Binding

CIBA requires the relying party to have an identifier for the user.

If this identifier is static and easily discoverable then there the chance that an attacker can fraudulently start a CIBA session

With UK Open Banking we've been thinking of a number of solutions to this:

THE CIBA FLOW

RP gains id_token by another means, e.g. via a traditional redirect flow.

This means that the RP has an identifier that is scoped to the RP AND that the context between the RP, OP and user has been established. The RP should also have informed the user when requesting authorisation via the redirect flow that it would use the granted identifier for these type of flows

Depending on the granularity of the authorisation being sought, there may be no need for a binding message.

The id_token can be associated with a users account at the RP and could also be linked to a physical identifier such as a card.

Best Decoupled experience Kiosk :

PSU has linked club card with the customer ID from ASPSP



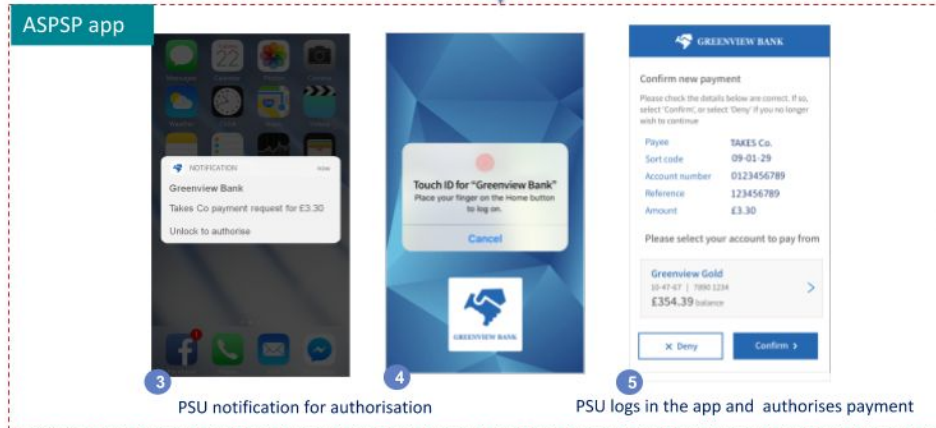
1 Clubcard has been linked to ASPSP customer ID



2 PSU scans their clubcard at the kiosk



6 Kiosk receives Payment authorisation



3 PSU notification for authorisation

4 PSU logs in the app and authorises payment



THE CIBA FLOW

RP requests unique identifier for a CIBA session from the user. The user obtains this from the OP via the OPs “app”

While degrading the user experience, this option strongly binds the sessions on the authentication and consumption devices.

There is no need for a binding message.

There is a requirement that the consumption device and the authentication device can communicate - either via NFC or QR Code.

DISCUSSION QUESTIONS

1. Given the real world requirement for decoupled flows, is CIBA the best fit?

2. How can the CIBA protocol be improved from a security perspective?

3. What are the additional security considerations in decoupled flows

THANKS!

@davidgtonge