CLIENT INITIATED BACKCHANNEL AUTHENTICATION

CIBA?

CIBA is an authentication flow like OpenID Connect. However, unlike OpenID Connect, there is direct Relying Party to OpenID Provider communication **without redirects** through the user's browser.

This specification has the concept of a **Consumption Device** (on which the user interacts with the Relying Party) and an **Authentication Device** (on which the user authenticates with the OpenID Provider and grants consent).

This specification allows a Relying Party that has **an identifier** for a user to obtain tokens from the OpenID Provider. The user starts the flow with the Relying Party at the Consumption Device, but authenticates and grants consent on the Authentication Device.

BRIEF HISTORY

gonz...@telefonica.com committed **087439f**
2016-08-10

Initial commit for Server Initiation Flow

Initial review of MODRNA Client initiated Backchannel Authentication Flow 1.0

Dave Tonge <dave.tonge@momentumft.co.uk> Fri, 26 May 2017, 12:59
to Openid-specs

Brian Campbell committed **659784f**
2018-12-11

rename the CIBA core doc to openid-client-initiated-backchannel-authentication-core.xml (from draft-mobile-client-initiated-backchannel-authentication.xml)

[Openid-specs-mobile-profile] Implementer's Draft of OpenID Connect Client Initiated Backchannel Authentication (CIBA) Core Approved

Mike Jones via Openid-specs-mobile-profile <openid-specs-mo... Tue, 5 Feb, 01:04
to openid-specs-mobile-profile@lists.openid.net

WHY DECOUPLED

For when the AUTHENTICATION device is not the CONSUMPTION device.

1. Granting authorisation to remote call centre agent
2. Using the strongly authenticated session on a smart device to grant authorisation to another device that is input constrained, or doesn't belong to the user.

WHY CIBA

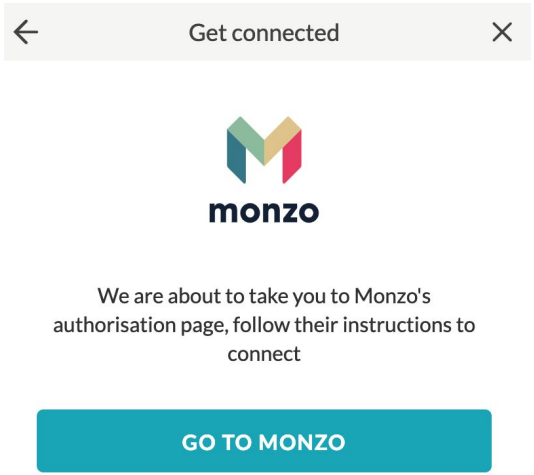
Many decoupled flows are possible using existing redirect based flows.

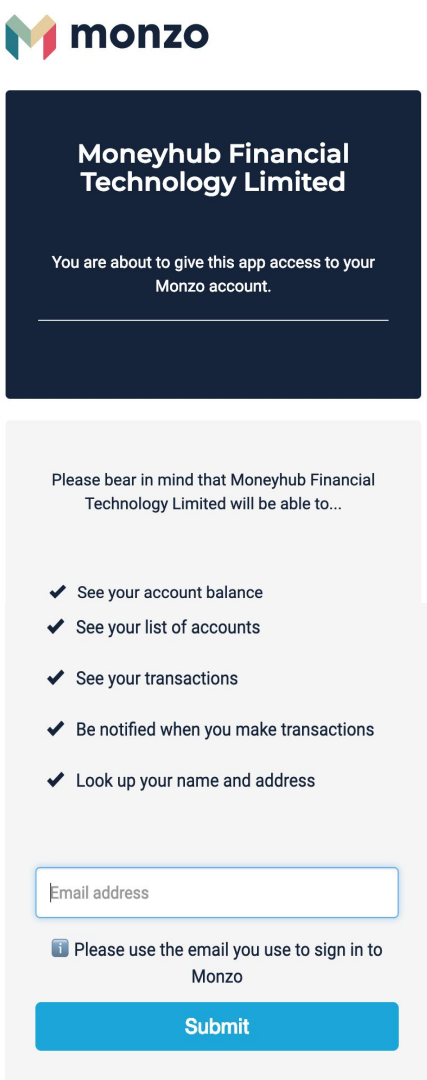
CIBA should not be used as a shortcut

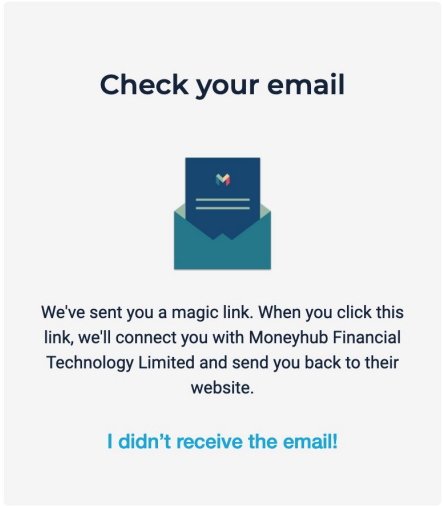
CIBA provides no way to cryptographically bind the session on the authentication device to the session on the consumption device.

BUT - CIBA is better than some of the ways decoupled is already implemented.

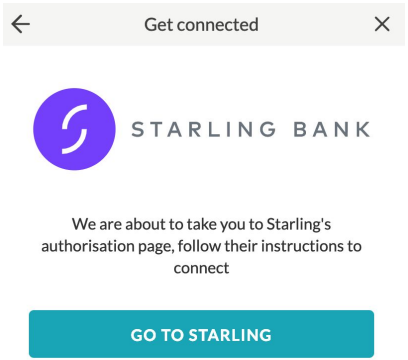
DECOUPLED IN THE WILD

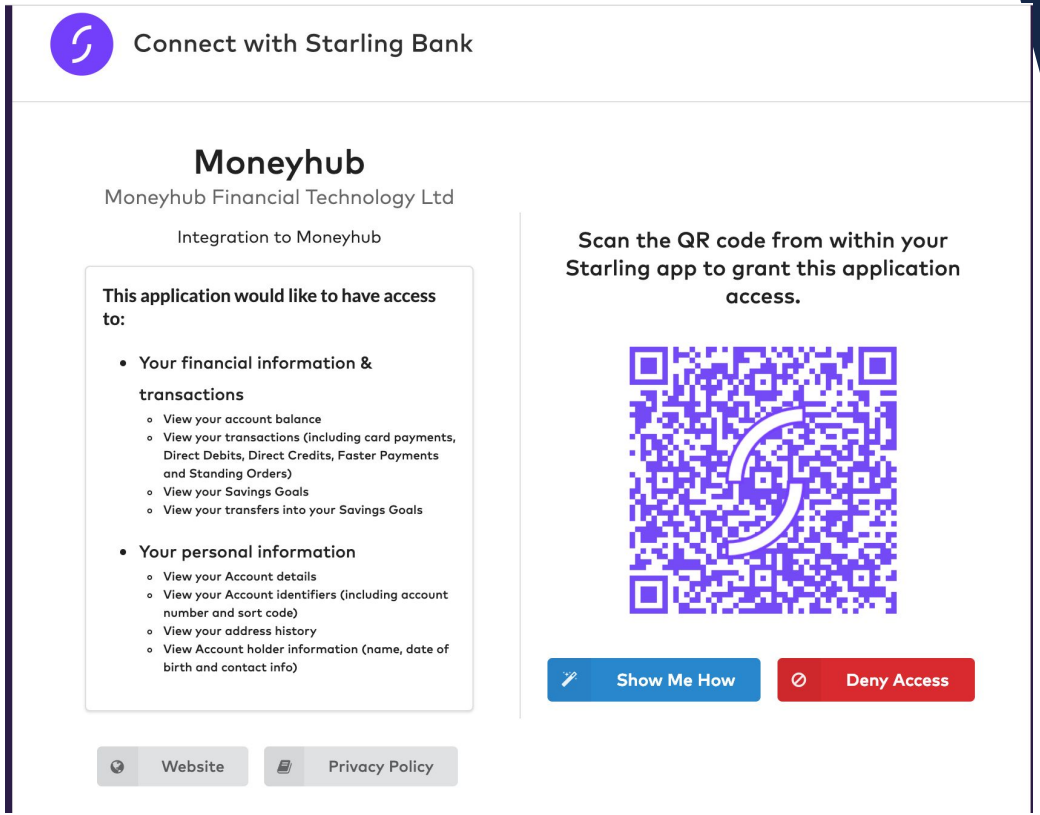






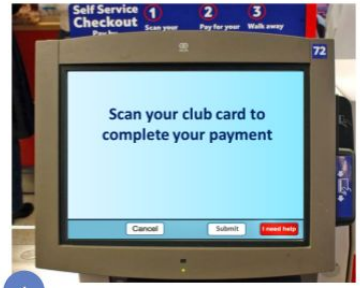
DECOUPLED IN THE WILD





CIBA FOR PAYMENTS

Best Decoupled experience Kiosk : **PSU has linked club card with the customer ID from ASPSP**

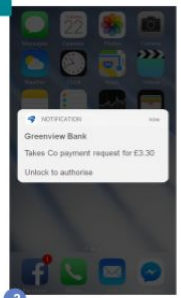


1 Clubcard has been linked to ASPSP customer ID

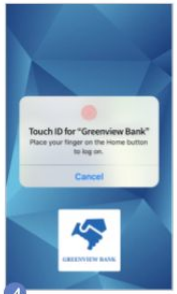
2 PSU scans their clubcard at the kiosk

6 Kiosk receives Payment authorisation

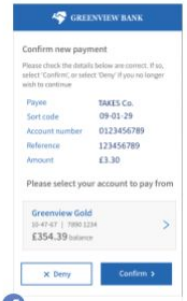
ASPSP app



3 PSU notification for authorisation



4



5 PSU logs in the app and authorises payment

THE CIBA FLOW

Back-channel

1. RP to OP: user123 wants to grant access to me

Front-channel

2. OP to user123: do you grant access to RP?
3. user123 to OP: yep

Back-channel

4. OP to RP: here is a token that allows you access for user123

CIBA MODES

POLL

RP polls the token endpoint

PING

OP sends a notification to the RP

RP gets tokens from token endpoint

PUSH

OP pushes tokens to the RP

AUTHENTICATION REQUEST

```
POST /bc-auth HTTP/1.1
   Host: server.example.com
   Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
   Content-Type: application/x-www-form-urlencoded

   scope=openid%20email%20example-scope&
   client_notification_token=8d67dc78-7faa-4d41-aabd-67707b374255&
   binding_message=W4SCT&
   login_hint_token=eyJ.....Ahawe7IPQ
```

AUTHENTICATION RESPONSE

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

{
  "auth_req_id": "1c266114-a1be-4252-8ad1-04986c5b9ac1",
  "expires_in": 3600,
  "interval": 2
}
```

PING CALLBACK

```
POST /cb HTTP/1.1
    Host: client.example.com
    Authorization: Bearer 8d67dc78-7faa-4d41-aabd-67707b374255
    Content-Type: application/json

    {
     "auth_req_id": "1c266114-a1be-4252-8ad1-04986c5b9ac1"
    }
```

TOKEN REQUEST

```
POST /token HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW

grant_type=urn%3Aopenid%3Aparams%3Agrant-type%3Aciba
  &auth_req_id=1c266114-a1be-4252-8ad1-04986c5b9ac1
```

TWO PROBLEMS

Session Binding

How do you ensure that the user at the authentication device is granting access to the correct consumption device?

Identification

What user identifier does the relying party use and how does it obtain it?

IDENTIFICATION

Four options

- **Discovery** - this works well with MNOs
- **Static Identifier** - open to abuse
- **Dynamic single-use identifier** - generated by the OP, this also solves the binding problem
- Previously issued **ID Token** - which could have been received via a redirect flow

All options supported by CIBA (`login_hint_token`, `id_token_hint` & `login_hint`)

SESSION BINDING

Three options

- Use a **dynamic single-use identifier**
- Let the **user** decide - If there is enough context on the authorisation being sought
- **Binding message** - displayed on the consumption device, verified by the user on the authentication device

ID TOKEN HINT

OAuth is rarely one-time use. Using an ID Token as a hint for CIBA provides a nice balance between usability and privacy.

1. Get an ID Token via a redirect flow
2. The ID Token binds the user's account at the OP with the user's account at the RP
3. When the user identifies herself at the RP, the RP can use the previously issued ID Token to start a CIBA flow

THANKS!

@davidgtonge

Moneyhub