Native Apps UX

OAuth for Native Mobile Apps Today
Log In to Example App

Manage your account, check notifications, comment on videos, and more.

- Use phone / email / username
- Continue with Facebook
- Continue with Apple
- Continue with Google
- Continue with Twitter
- Continue with Instagram

By continuing, you agree to our Terms of Service and acknowledge that you have read our Privacy Policy to learn how we collect, use, and share your data.

Don't have an account? Sign up
2-Step Verification
To help keep your account safe, Google wants to make sure it's really you trying to sign in

☑ aaron.parecki@gmail.com

You're all set

☐ Don't ask again on this device
Log In to Example App

Manage your account, check notifications, comment on videos, and more.

- Use phone / email / username
- Continue with Facebook
- Continue with Apple
- Continue with Google
- Continue with Twitter
- Continue with Instagram

By continuing, you agree to our Terms of Service and acknowledge that you have read our Privacy Policy to learn how we collect, use, and share your data.

Don't have an account? Sign up
Great for third party apps

- Secure isolation between app and system browser
- Leverages existing session at the OAuth server
- Supports phishing-resistant MFA
Developers want a better user experience for first-party apps
What is happening today

People are finding workarounds to avoid RFC8252

- Custom DIY solutions for native apps
- Using Resource Owner Password Grant
  - (Unable to add MFA)
- OAuth servers creating proprietary APIs to facilitate direct interaction with native apps
- Scripting hidden web views to emulate user interaction with the AS
What is happening today

All of these lead to worse outcomes
Authorization Code Flow for Web Apps

- **OAuth Client**
  - Redirect to AS
  - `client_id`, `scope`, `state`, `code_challenge`, etc

- **Authorization Server**
  - Token Request (Back Channel)
  - `code`, `state`

- **OAuth Client**
  - Redirect to Client

- **Browser (User Agent)**
Authorization Code Flow for Web Apps

"Out of Scope" of OAuth specs:
- Registration
- Authentication
- MFA
- Consent gathering
- Risk assessment
- etc. etc.

Blue arrows are the OAuth Authorization Code Flow
Authorization Code Flow for Native Apps

Authorization Server

Implementation-Specific
- Registration
- Authentication
- MFA
- Consent gathering
- Risk assessment
- etc. etc.

OAuth Client (Native App)

Blue arrows are the new flow

code, state

Token Request

client_id, scope, state, code_challenge, etc.
Goals

- Reuse existing OAuth building blocks as much as possible
- Mirror the web authorization code flow, defining how the client starts and ends the flow
  - Leave the specifics of the user authentication out of the core framework
- Specifics of user authentication can be proprietary to an AS as they are today, or can be defined as extensions
Authorization Challenge Endpoint

- New endpoint
- Accepts parameters that would have been included in the query string to the authorization endpoint
  - including any extensions like PKCE, Resource Indicators, OpenID Connect
- Accepts POST from client to start and continue an authorization
  - The AS defines what the client sends in the requests and defines its own error codes
- Response is an authorization code or error
Token Endpoint

- No changes to the token request
- Client POSTs the directly-obtained authorization code to get an access token
Authorization Challenge Endpoint

Why a new endpoint?

- Existing authorization endpoint is never interacted with by the OAuth client today, only by the browser
- It expects to receive requests from a **User Agent**, and return **HTML**
- Initial feedback has indicated people are unwilling to modify this behavior to accept a direct POST from a client and return **JSON**
Error Responses

Token Endpoint Error Response

- Any request to the token endpoint (e.g. with refresh token) can fail with an error indicating the client needs to obtain a fresh authorization from the user and start the flow over

Resource Server Error Response

- No changes in this draft
- Can use Step-Up Authentication to tell the client to start a new flow

Next: To be determined…
To Be Determined…

- Rename “device session” (#27)
  - This is not really a device session, it’s a handle to the authorization session
  - Maybe “auth_req_id” like CIBA, or “auth_req_txn” if it has different semantics from CIBA

- Enable transitioning to the web (#16)
  - In some cases, the authorization server may want to require the user bounces to the web, even in mobile
  - Ideally the web context can resume with any existing context from the app session to avoid the UX appearing like it’s a fresh start on the web
To Be Determined…

- We’ve heard a lot of people wanting to do FIDO as an OAuth grant, how should we enable that within this framework?
- Is there interest in creating more specific profiles of this for current authentication mechanisms such as FIDO