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

You're all set

☑ Don't ask again on this device

Next
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 → Authorization Server
(client_id, scope, state, code_challenge, etc)

Authorization Server → OAuth Client
(code, state)

Browser (User Agent)

Redirect to AS

Redirect to Client
Authorization Code Flow for Web Apps

- **OAuth Client**
  - client_id, scope, state, code_challenge, etc.

- **Browser (User Agent)**

- **Authorization Server**

- **OAuth Client**
  - code, state

- **Redirect to AS**

- **Token Request (Back Channel)**

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

Blue arrows are the OAuth Authorization Code Flow
Native Apps Flow

Blue arrows are the new flow

Authorization Server

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

Native App

client_id, scope, state, code_challenge, etc.

code, state

Token Request
Questions

● Is OAuth WG the right place for this?
  ○ The artifacts issued will still be access tokens and refresh tokens, and likely ID tokens as well

● Should the flow leave the details of the app-AS exchange unspecified to mirror the Authorization Code flow?
  ○ Should the flow define a “challenge-response” format that can be extended with provider-specific logic?