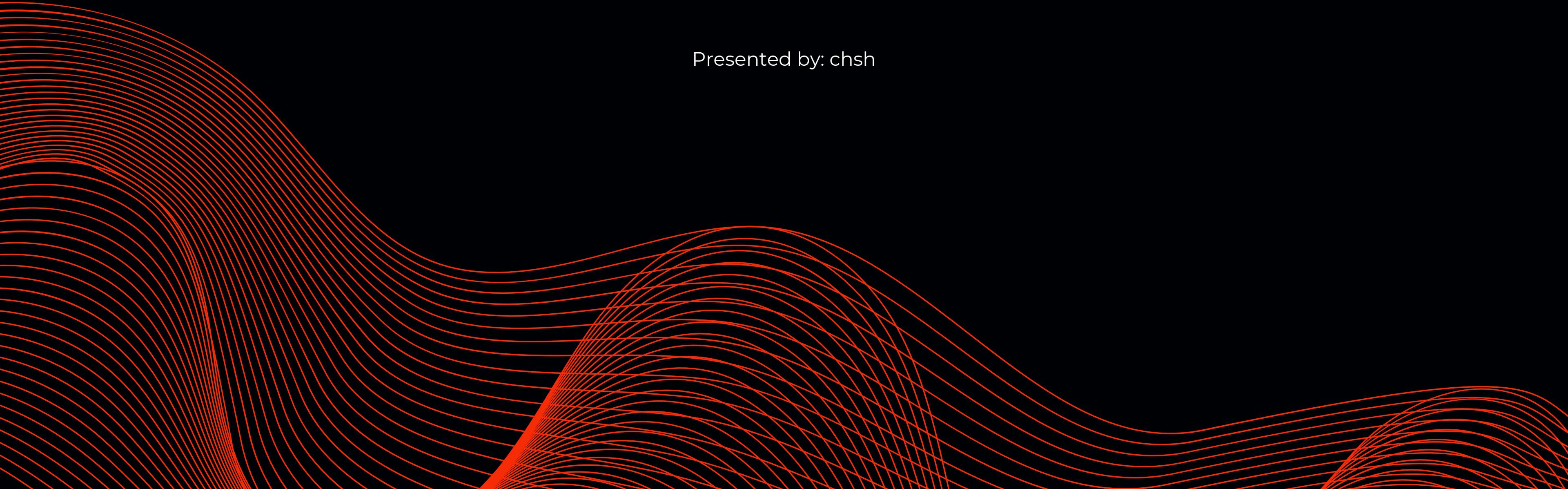


# **YOU ONLY LOGIN ONCE**

Presented by: chsh



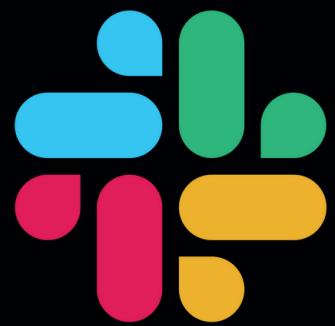
# SINGLE SIGN ON



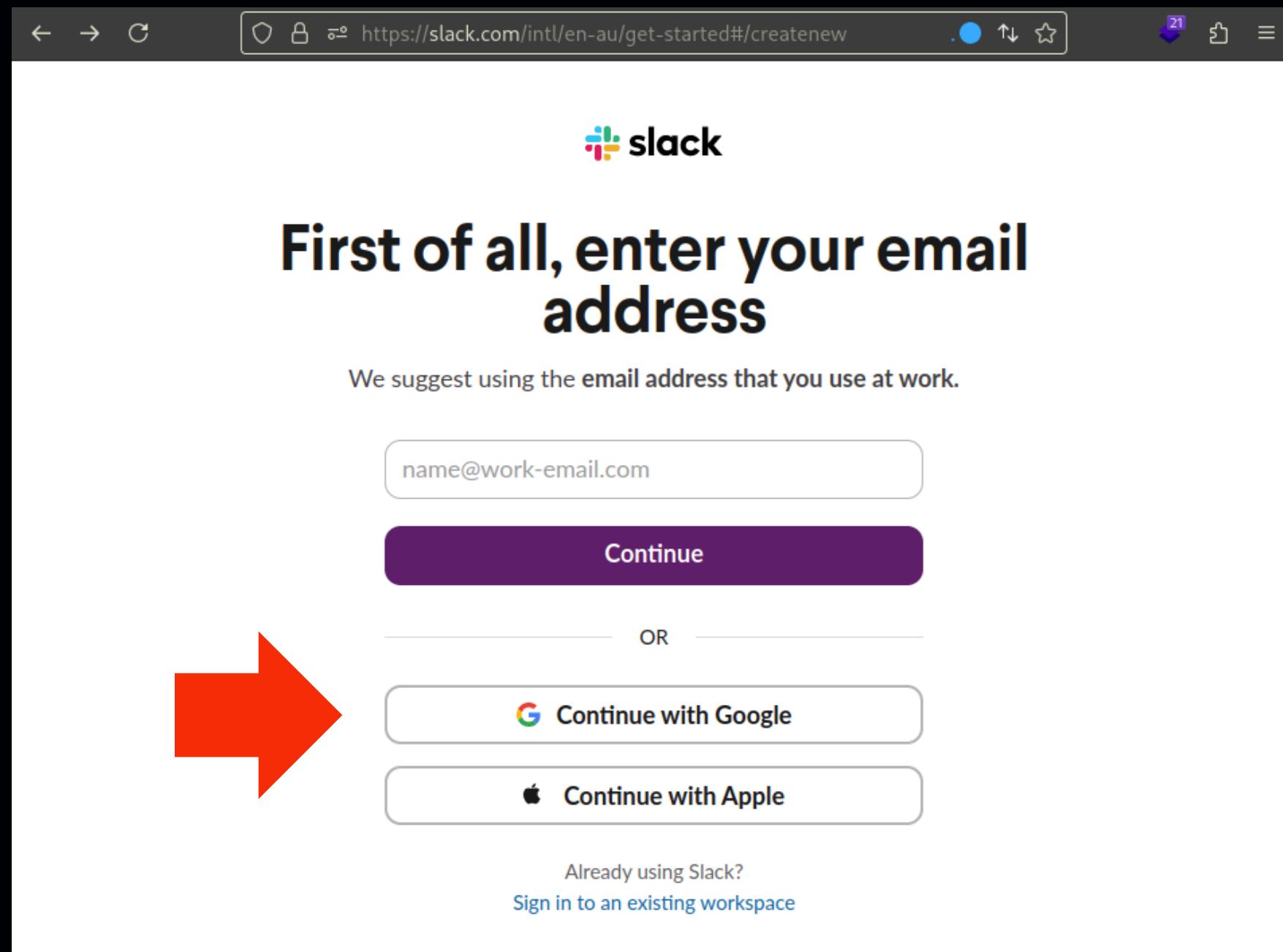
# CONTENT

- OpenID Connect protocol (OIDC)
- OIDC bug class that I think is cool
- OIDC bug that I saw in the wild (less cool)

# LOGGING IN TO SLACK



<https://slack.com>



[https://accounts.google.com/...](https://accounts.google.com/)

← → C [https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606...](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606...) ↻ ⌂ ↑ ☆ ☰

 Sign in with Google



## Choose an account

to continue to [Slack](#)

Before using this app, you can review Slack's [privacy policy](#) and [terms of service](#).

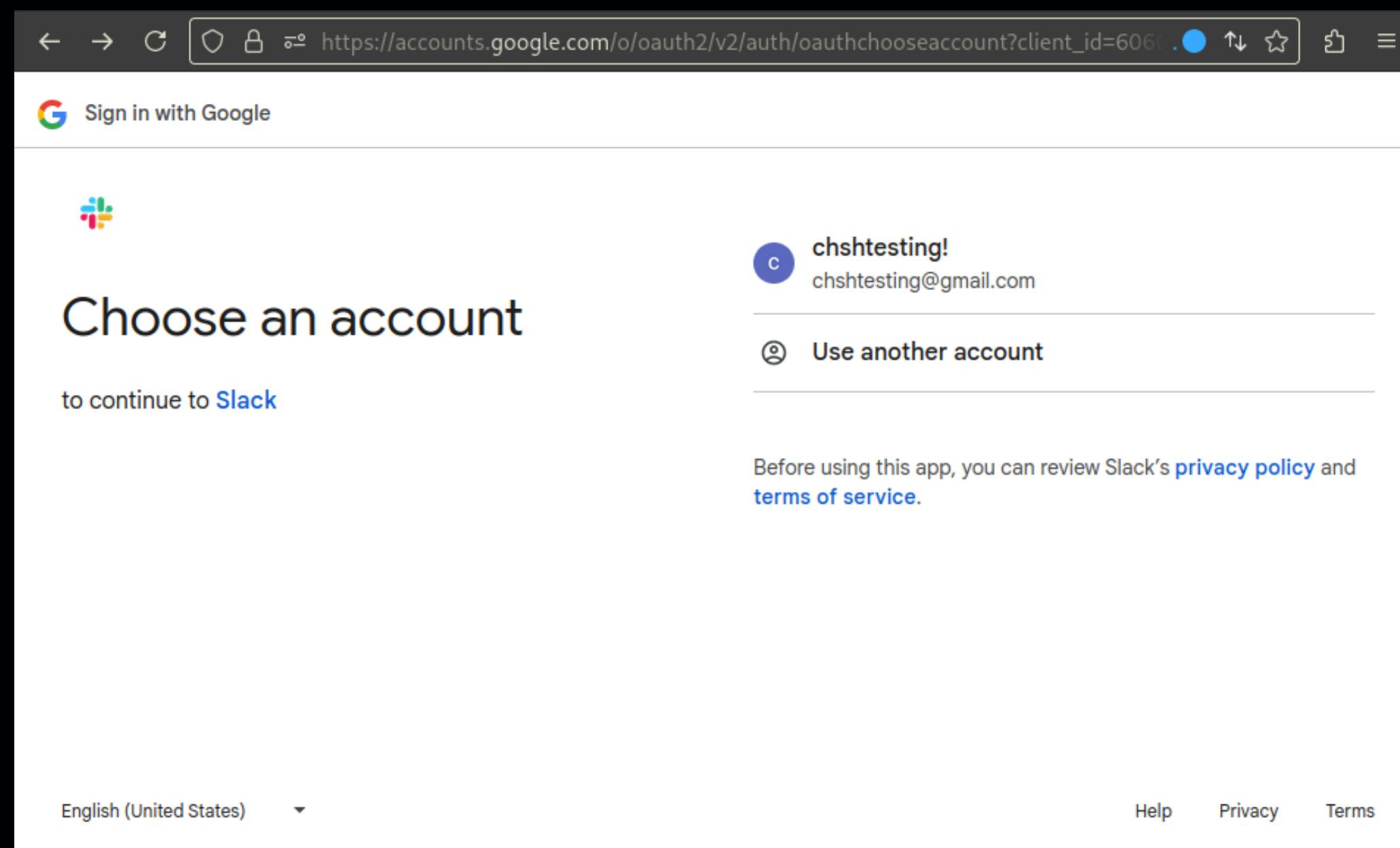
chshtesting!  
chshtesting@gmail.com

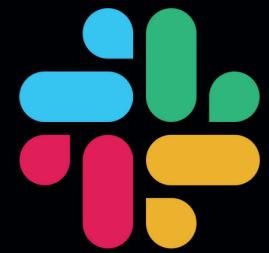
Use another account

English (United States) ▾

Help Privacy Terms

[https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state=%7B%22provider%22%3A%22google%22,%22origin%22%3A%22signin%22,%22timestamp%22%3A1724683119,%22visitor%22%3A%22796e18fba4f59969ef46afe47b252c14%22%7D%7C0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid%20email%20profile&response_type=code&access_type=offline&state=%7B%22provider%22%3A%22google%22,%22origin%22%3A%22signin%22,%22timestamp%22%3A1724683119,%22visitor%22%3A%22796e18fba4f59969ef46afe47b252c14%22%7D%7C0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow)





## Relaying Party

[https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response_type=code&access_type=offline&state={)

A screenshot of a web browser window displaying the Google OAuth sign-in page for Slack. The URL in the address bar is [https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response_type=code&access_type=offline&state={). The page title is "Sign in with Google". The main content is titled "Choose an account" and says "to continue to Slack". It shows a list of accounts: "chshtesting!" (chshtesting@gmail.com) and "Use another account". A note at the bottom states: "Before using this app, you can review Slack's [privacy policy](#) and [terms of service](#)". The bottom of the page includes links for "English (United States)" (dropdown), "Help", "Privacy", and "Terms".

← → C [https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response_type=code&access_type=offline&state={)

G Sign in with Google

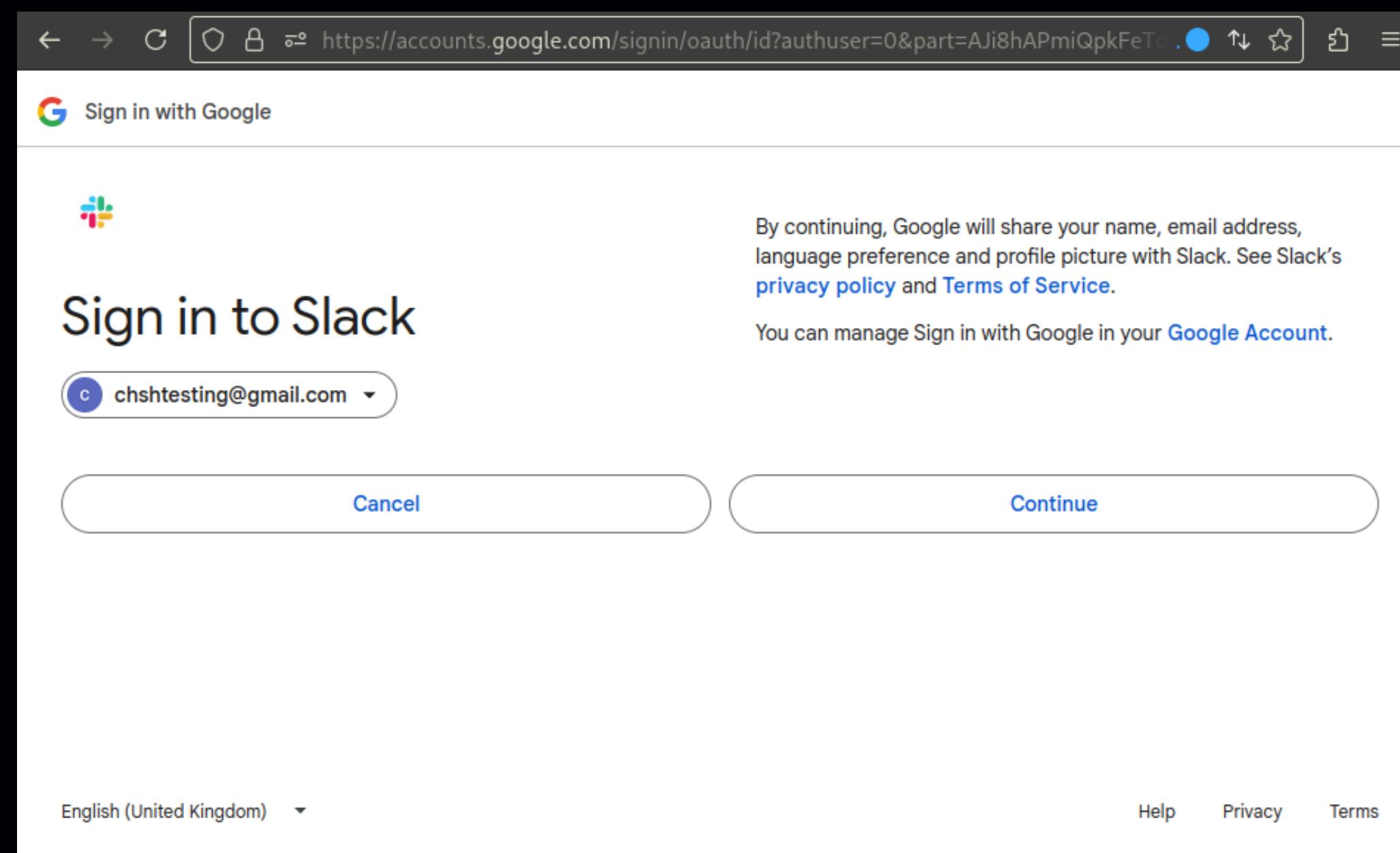
Choose an account

to continue to Slack

Before using this app, you can review Slack's [privacy policy](#) and [terms of service](#).

English (United States) ▾ Help Privacy Terms

[https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid%20email%20profile&response_type=code&access_type=offline&state={)



[https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response_type=code&access_type=offline&state={)

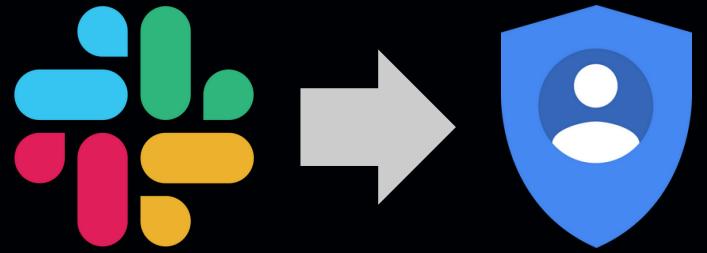
The screenshot shows a browser window with the Slack logo at the top. Below it is the developer tools Network tab interface. A tooltip is overlaid on the table, providing details about the requests.

Status	Method	Domain	File	Initiator	Type	Transferred	Size	Time
200	POST	play.google.com	login?format=json&last=true&authuser=0	fetch	JS BINDING RD...	0 B	0 ms	40.96 s
302	POST	accounts.google.com	id?as=S1679627720:1724683146029109&authuser=0&	m=n73qwf,SCuO...	html	38.67 kB	3...	212 ms
302	GET	accounts.google.com	consent?as=S1679627720:1724683146029109&authus	document	html	37.03 kB	3...	246 ms
0	POST	accounts.google.com	browserinfo?f.sid=8456323435363838408&bl=boq_id	m=_b,_tp:328 (xhr)				Blocked By uBloc...
302	GET	oauth2.slack.com	end?state={"provider":"google","origin":"signin","time	document	html	37.52 kB	3...	546 ms
200	GET	slack.com	signin					
	GET	a.slack-edge.com	rollup-sla					
	GET	a.slack-edge.com	ontrust_					
	GET	a.slack-edge.com	lato-2-co					
	GET	a.slack-edge.com	_generic.					
	GET	a.slack-edge.com	rollup-sla					
	GET	a.slack-edge.com	rollup-sla					
200	GET	a.slack-edge.com	manifest.					

**Original:** <https://oauth2.slack.com/signin/oauth/google/end?state=%7B%22provider%22%3A%22google%22%2C%22origin%22%3A%22signin%22%2C%22code%22%3A%2240e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&code=4www.googleapis.com/auth/userinfo.profile+https://www.googleapis.com/auth/userinfo.email&authuser=0&prompt=consent>

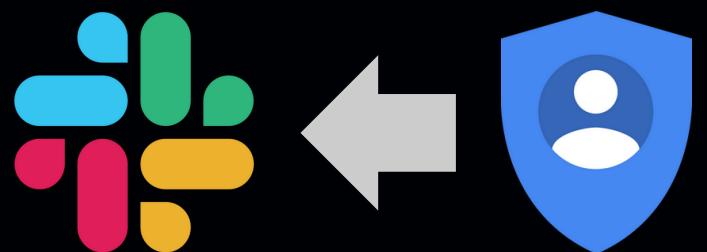
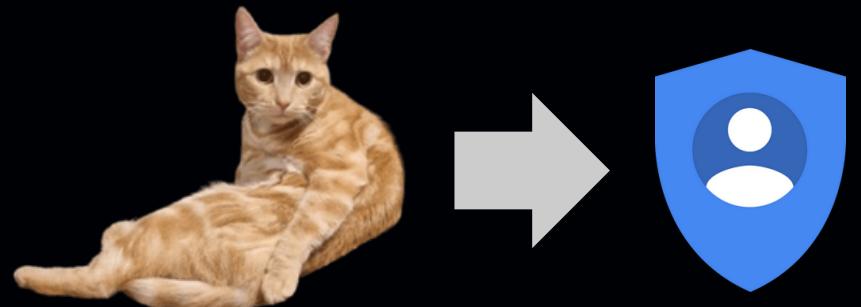
**Decoded:** [https://oauth2.slack.com/signin/oauth/google/end?state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&code=4www.googleapis.com/auth/userinfo.profile+https://www.googleapis.com/auth/userinfo.email&authuser=0&prompt=consent](https://oauth2.slack.com/signin/oauth/google/end?state={)

43 requests | 1.45 MB / 1.17 MB transferred | Finish: 36.59 s | DOMContentLoaded: 3.92 s | load: 4.08 s



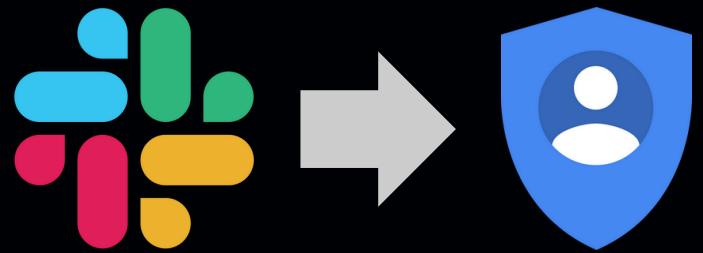
[https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadb3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid%20email%20profile&response_type=code&access_type=offline&state={)

*Authentication request*



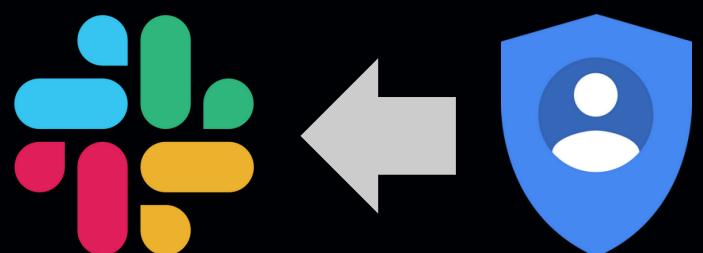
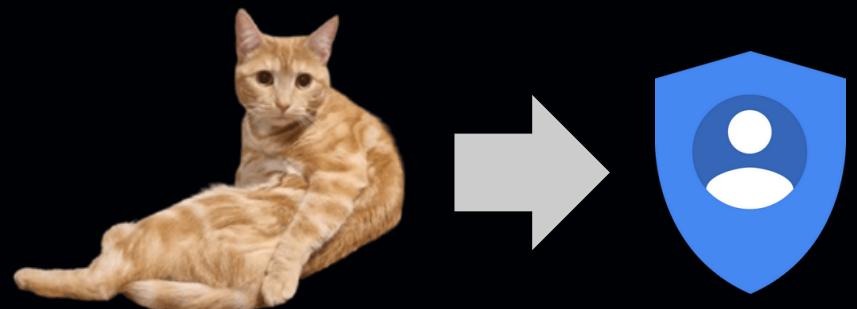
[https://oauth2.slack.com/signin/oauth/google/end?state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadb3c3a1&code=4/0AQIEd8ybjf6rEDd5FsglnCIFI9Pg0qF5vU1yTYBuuaG49WJnkukVIQBaVCdVFMgV7Ck2eg&scope=email%20profile open https://www.googleapis.com/auth/userinfo.profile https://www.googleapis.com/auth/userinfo.email&authuser=0&prompt=consent](https://oauth2.slack.com/signin/oauth/google/end?state={)

*Authentication response*



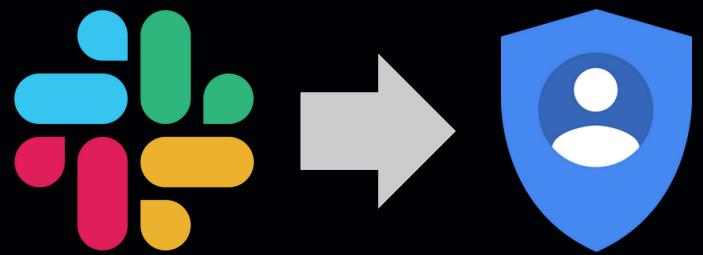
[https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=code&access\\_type=offline&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadb3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid%20email%20profile&response_type=code&access_type=offline&state={)

*Authentication request*



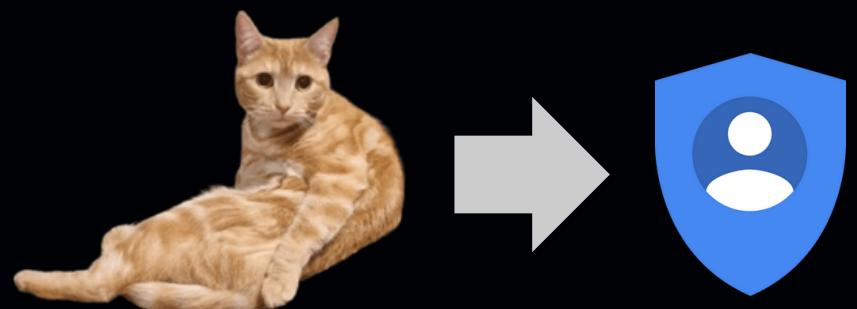
[https://oauth2.slack.com/signin/oauth/google/end?state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadb3c3a1&code=4/0AQIEd8ybjf6rEDd5FsglnCIFI9Pg0qF5vU1yTYBuuaG49WJnkukVIQBaVCdVFMgV7Ck2eg&scope=email profile openid https://www.googleapis.com/auth/userinfo.profile](https://oauth2.slack.com/signin/oauth/google/end?state={)  
<https://www.googleapis.com/auth/userinfo.email&authuser=0&prompt=consent>

*Authentication response*



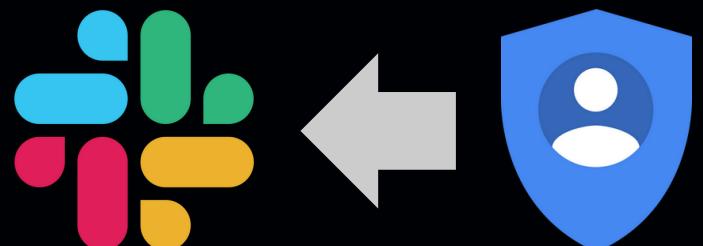
[https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client\\_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect\\_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response\\_type=id\\_token&nonce=1234&state={"provider":"google","origin":"signin","timestamp":1724683119,"visitor":"796e18fba4f59969ef46afe47b252c14"}|0e201ea7bbbed1ab29fba6000258397c02b391792d110740d86561bfadbf3c3a1&prompt=consent&service=Iso&o2v=2&ddm=0&flowName=GeneralOAuthFlow](https://accounts.google.com/o/oauth2/v2/auth/oauthchooseaccount?client_id=606092904014-s1u3idjanlbhr4ns5b1hcjgfn63cr9nh.apps.googleusercontent.com&redirect_uri=https://oauth2.slack.com/signin/oauth/google/end&scope=openid email profile&response_type=id_token&nonce=1234&state={)

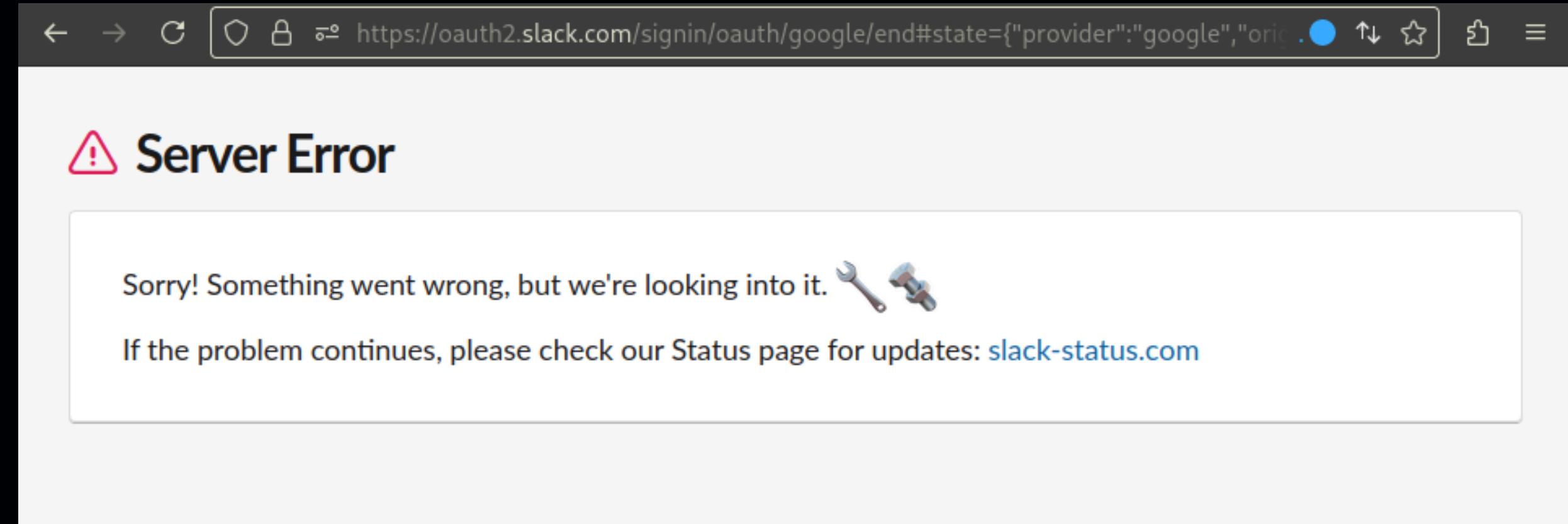
*Authentication request*



[https://oauth2.slack.com/signin/oauth/google/end#state={"provider":"google","origin":"signin","timestamp":1724764245,"visitor":"796e18fba4f59969ef46afe47b252c14"}|81cdfea6c1e8dacfe2b43e7672239ac847ef2f4f95008cb7f5e0b1cf6339c86a&id\\_token=eyJhbGciOiJSUzI1NilsImtpZCI6ImE0OTM5MWJmNTJiNThjMWQ1NjAyNTVjMmYyYTA0ZTU5ZTlyYTdiNjUiLCJ0eXAiOiJKV1QiQifQ.eyJpc3MiOiJodHRwczovL2FjY291bnRzLmdvb2dsZS5jb20iLCJhenAiOiI2M...&authuser=0&prompt=consent&version\\_info=CmxfU1ZJX0VPdmNxSm14bFlnREdCQWIQMDFCUIVSSVpsOXhRWGhNU1hsRWFXUIZZMVpLVGtrMGVITk9jVzISU2pWR2NFMTFVR2xoWmpWamN6UnhZVzlJTFVGME9XcEdTazFuV0UxcWVGTmphd18](https://oauth2.slack.com/signin/oauth/google/end#state={)

*Authentication response*





# “Non-happy path”

*-Frans Rosen, <https://labs.detectify.com/writeups/account-hijacking-using-dirty-dancing-in-sign-in-oauth-flows/#non-happy-paths-in-the-oauth-dance>*

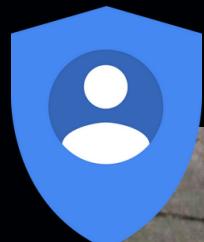
# OAuth 2.0 ← built on → OIDC

The screenshot shows the IETF RFC 6749 document page. At the top, it displays the document's metadata: Internet Engineering Task Force (IETF), Request for Comments: 6749, Obsoletes: 5849, Category: Standards Track, and ISSN: 2070-1721. The author is listed as D. Hardt, Ed. Microsoft, October 2012. Below the metadata, the title "The OAuth 2.0 Authorization Framework" is visible. The "Abstract" section describes the OAuth 2.0 protocol as enabling a third-party application to obtain limited access to an HTTP service. It mentions that this specification replaces and obsoletes the OAuth 1.0 protocol described in RFC 5849. The "Status of This Memo" section states that this is an Internet Standards Track document. The "Copyright Notice" section includes the copyright notice for IETF Trust and the document authors. The "Legal Provisions" section refers to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents.

The screenshot shows the OpenID Connect Core 1.0 incorporating errata set 2 document. At the top right, there is a "TOC" link. Below the header, a table lists the names and organizations of the contributors. The table has two columns: "Final" and "N. Sakimura". The contributors listed are N. Sakimura (NAT.Consulting (was at NRI)), J. Bradley (Yubico (was at Ping Identity)), M. Jones (Self-Issued Consulting (was at Microsoft)), B. de Medeiros (Google), C. Mortimore (Disney (was at Salesforce)), and the date December 15, 2023. The main content area features a red header "OpenID Connect Core 1.0 incorporating errata set 2". The "Abstract" section defines OpenID Connect 1.0 as a simple identity layer on top of the OAuth 2.0 protocol, enabling clients to verify the identity of the end-user and obtain basic profile information. The "Table of Contents" section lists the following chapters: 1. Introduction (1.1 Requirements Notation and Conventions, 1.2 Terminology, 1.3 Overview), 2. ID Token, 3. Authentication (3.1 Authentication using the Authorization Code Flow (3.1.1 Authorization Code Flow Steps, 3.1.2 Authorization Endpoint)).

# RECAP

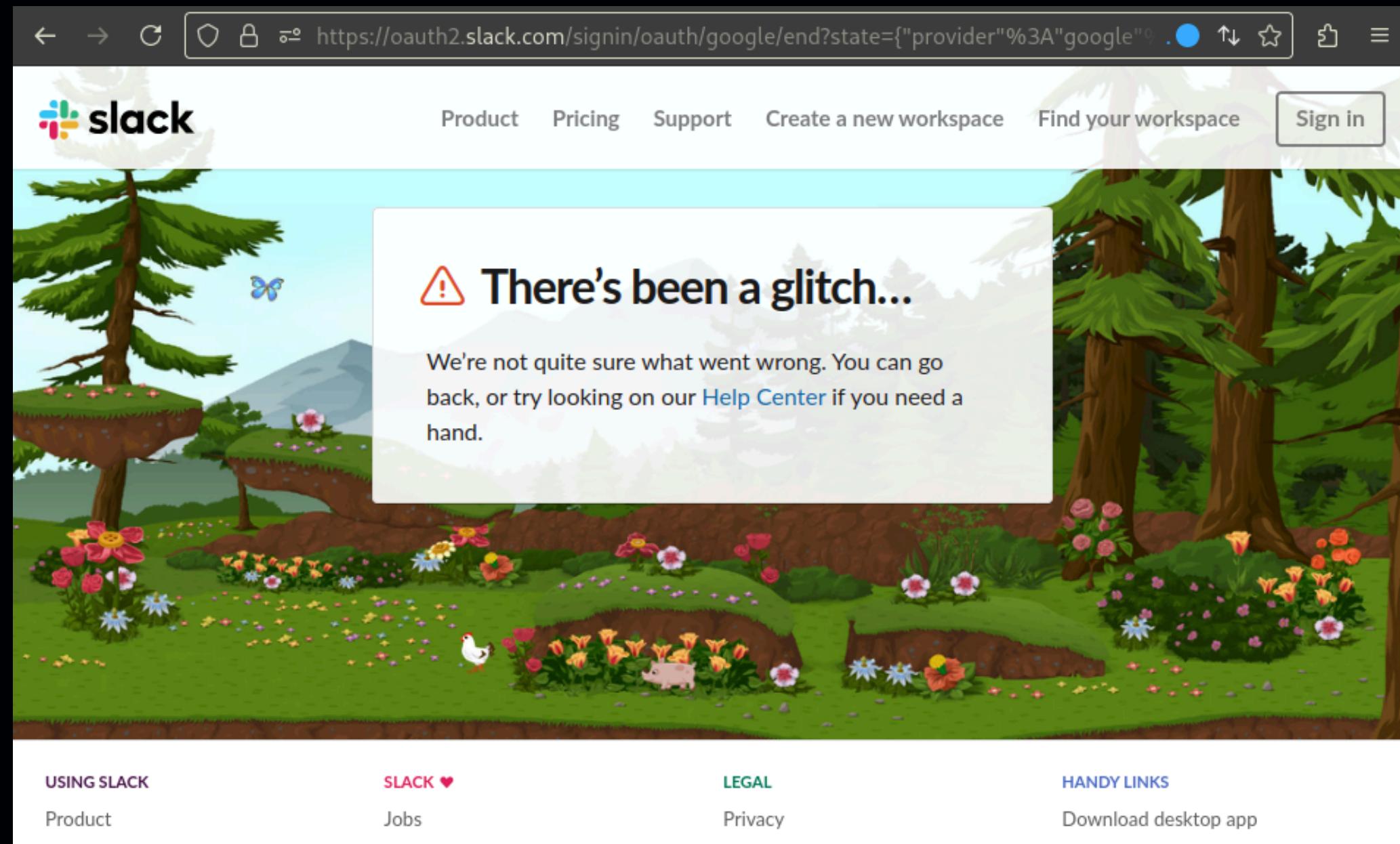
- OpenID Connect (OIDC) is an SSO protocol based on OAuth 2.0



Identity  
Provider



Relaying  
Party



A screenshot of a web browser displaying the Slack sign-in page at [https://slack.com/signin?error\\_user\\_oauth=error\\_user\\_oauth&provider=google#sign\\_in](https://slack.com/signin?error_user_oauth=error_user_oauth&provider=google#sign_in). The page features the Slack logo and navigation links for new users and account creation. It prominently displays two sign-in buttons: "Sign In With Google" and "Sign In With Apple". Each button is enclosed in a rounded rectangle and accompanied by a pink warning message about a connection issue.

← → ⌛ 🔒 https://slack.com/signin?error\_user\_oauth=error\_user\_oauth&provider=google#sign\_in ● ↑ ☆ ↴ ⌂ ≡

**slack**

New to Slack?  
[Create an account](#)

# Sign in to Slack

We suggest using the email address you use at work.

[Sign In With Google](#)

Something weird happened. Please try to sign up again.

[Sign In With Apple](#)

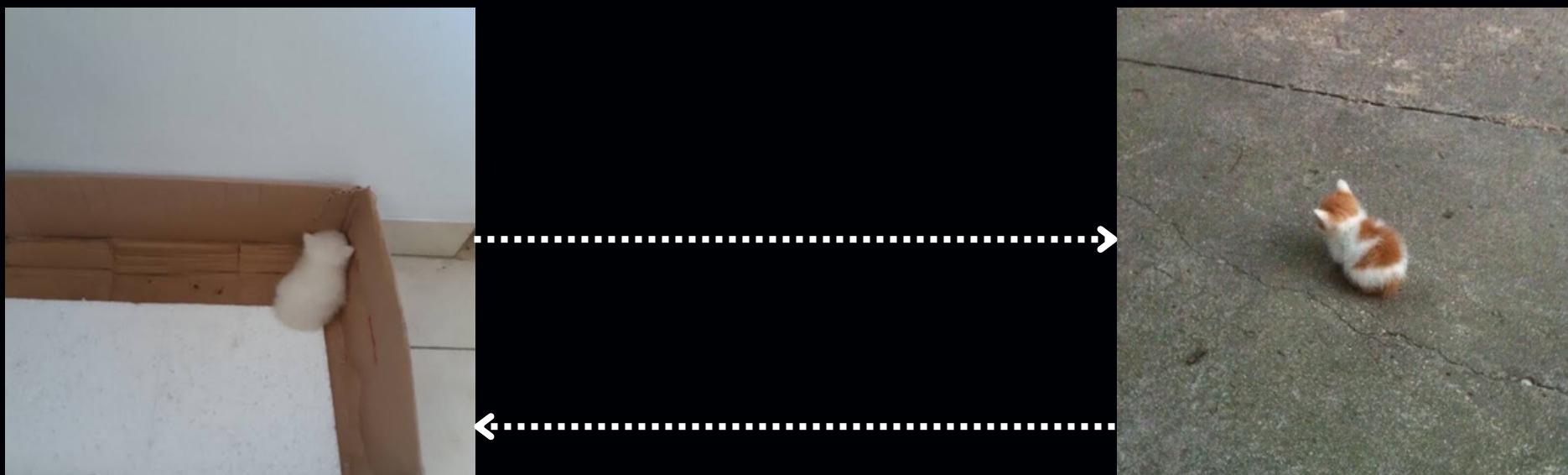
Something weird happened. Please try to sign up again.

# Non-happy paths

- When codes or tokens issued in the OAuth flow are not consumed by the page they land on and remain in the website URL
- Can be accessed by JavaScript using `location.href`
- To XSS... and beyond?

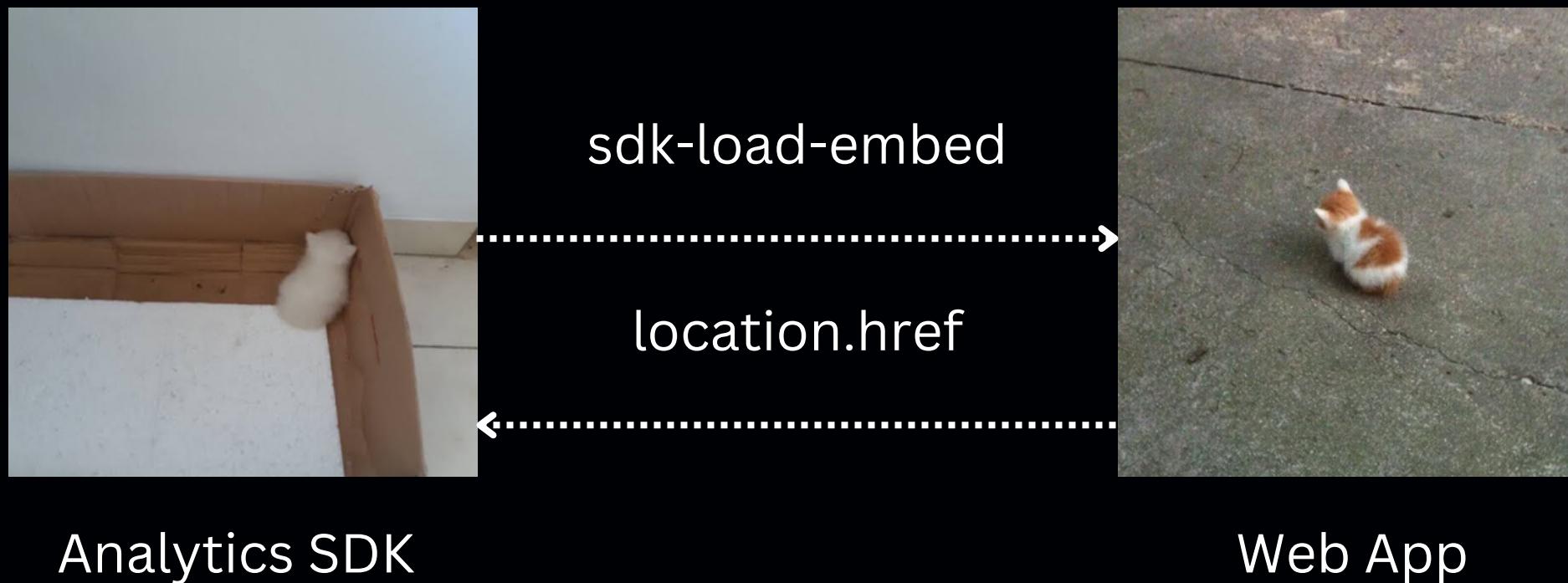
# postMessage()

- Allows cross origin communication between websites
- Should validate origin of sender for security
- Can be used to create URL-leaking “gadgets”



# postMessage()

- Allows cross origin communication between websites
- Should validate origin of sender for security
- Can be used to create URL-leaking “gadgets”



# Example attack

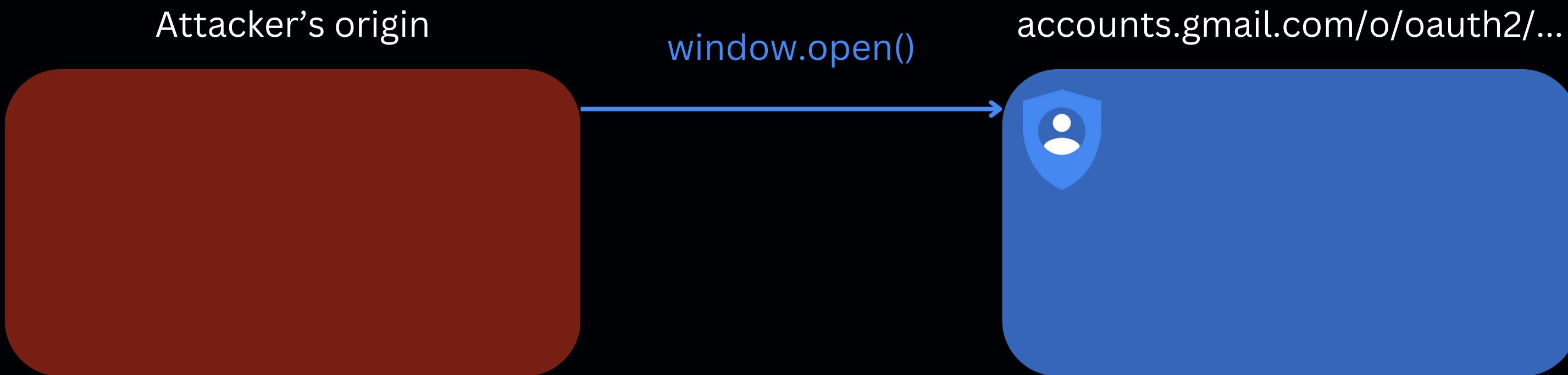
- Attacker creates a malicious page which is loaded in the target's browser

Attacker's origin



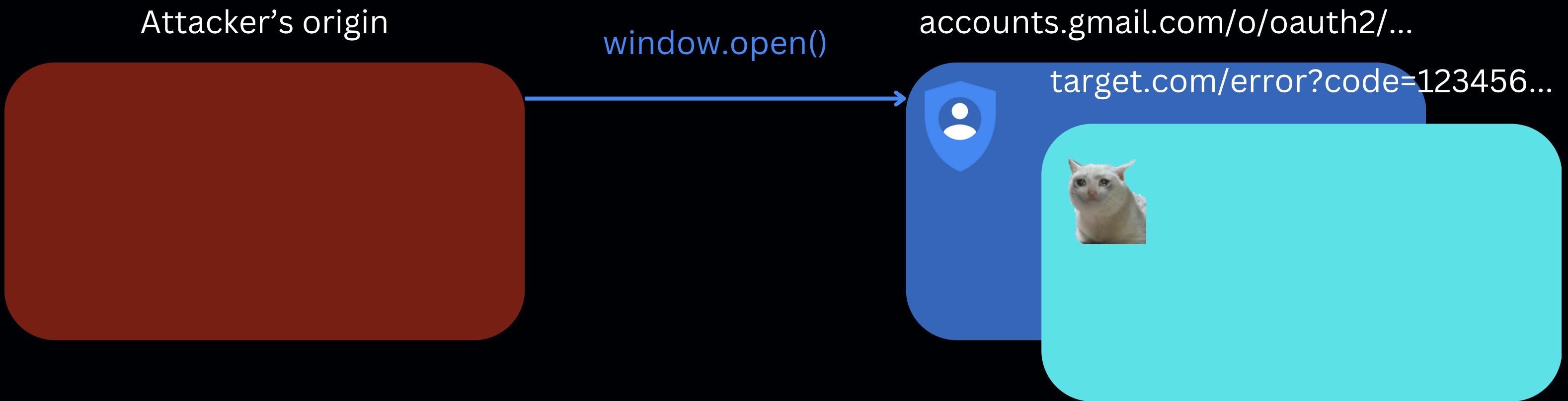
# Example attack

- Attacker creates a malicious page which is loaded in the target's browser



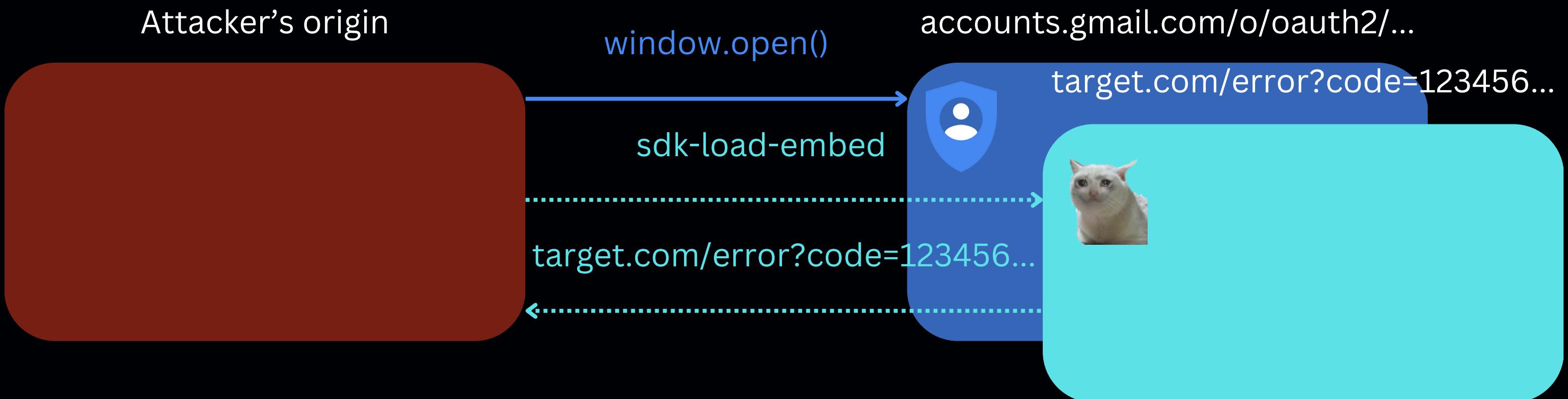
# Example attack

- Attacker creates a malicious page which is loaded in the target's browser



# Example attack

- Attacker creates a malicious page which is loaded in the target's browser





# WHAT IS OUT THERE



I wish  
vulnerabilities  
were real

# Custom flows?

- Saw an authentication request with the parameter  
response\_type=pi.flow

# Custom flows?

- Saw an authentication request with the parameter response\_type=pi.flow
- Custom flow defined by Ping identity

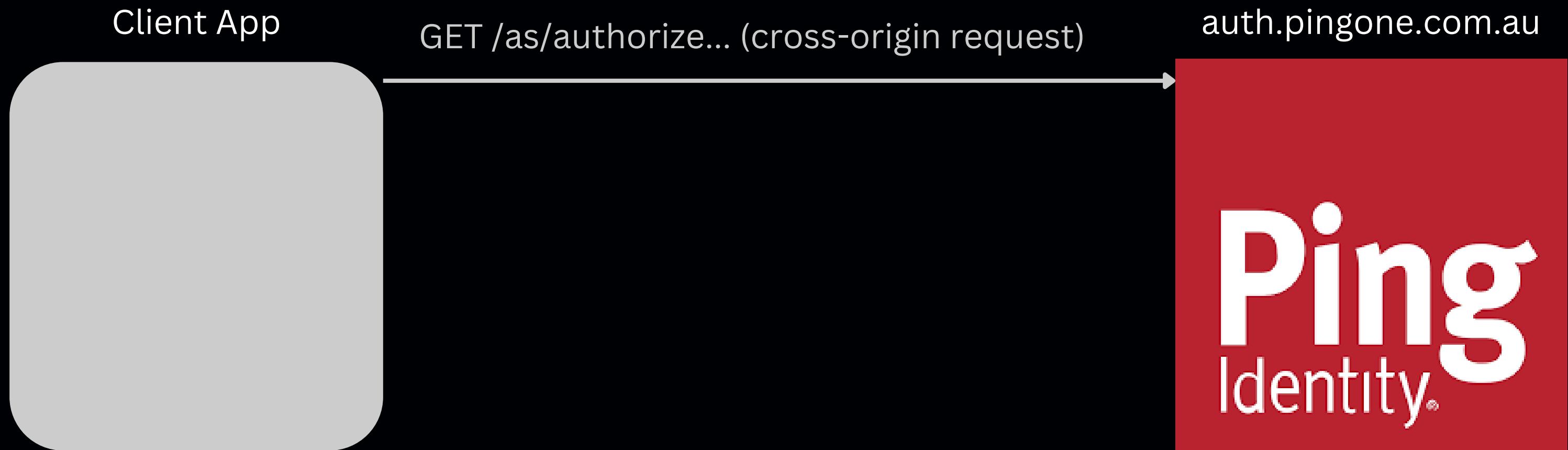


# Custom flows?

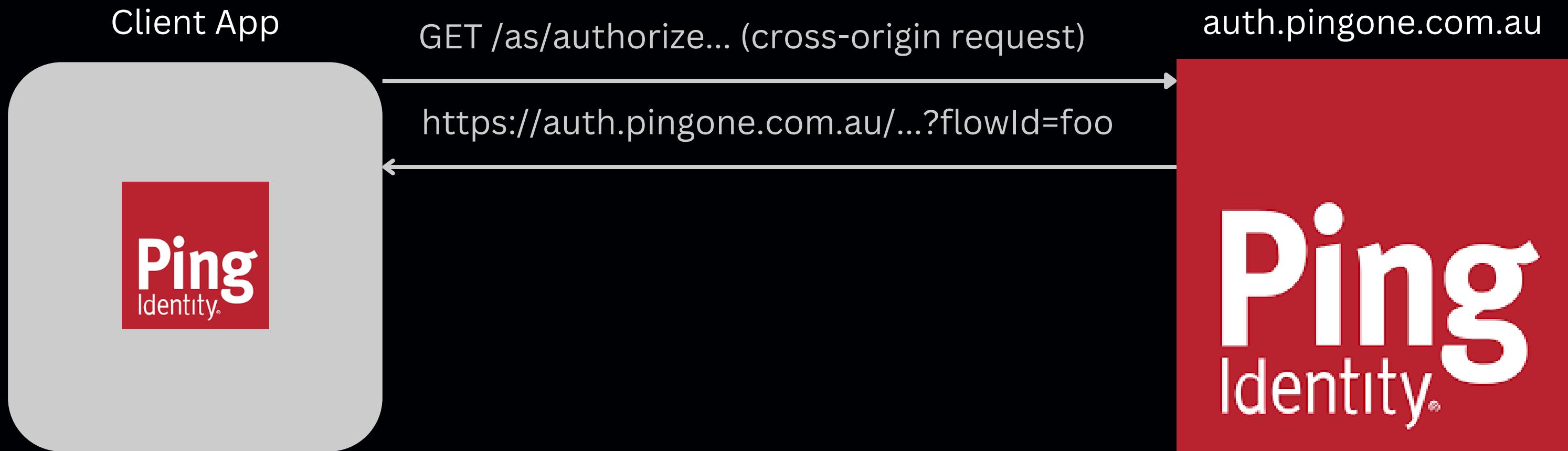
- Saw an authentication request with the parameter response\_type=pi.flow
- Custom flow defined by Ping identity
- No need for redirects between origins
- Code or token is returned within response to API



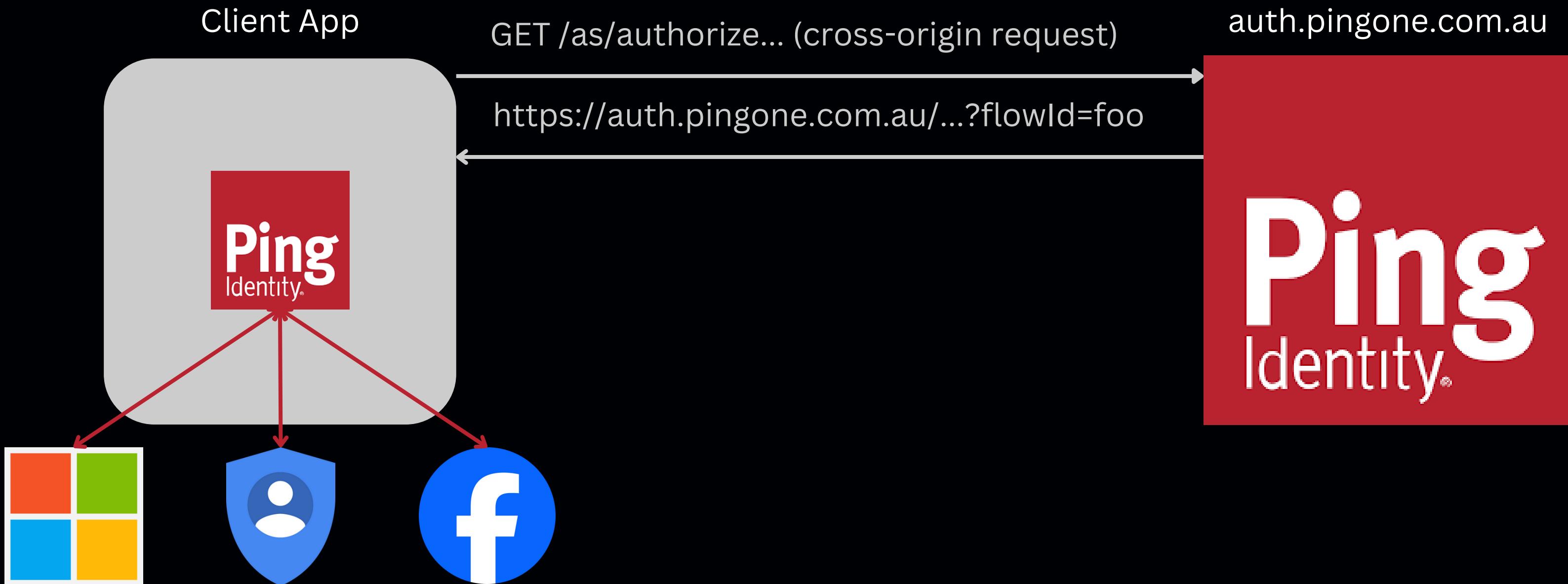
# Custom flows?



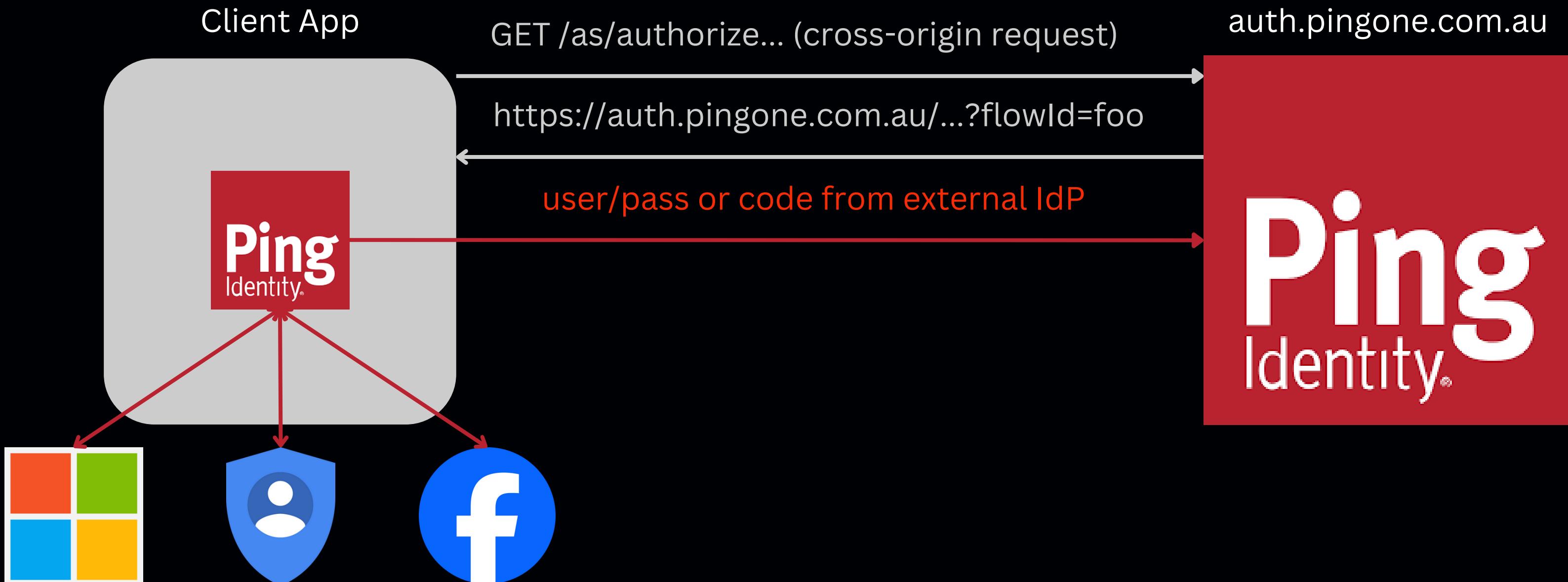
# Custom flows?



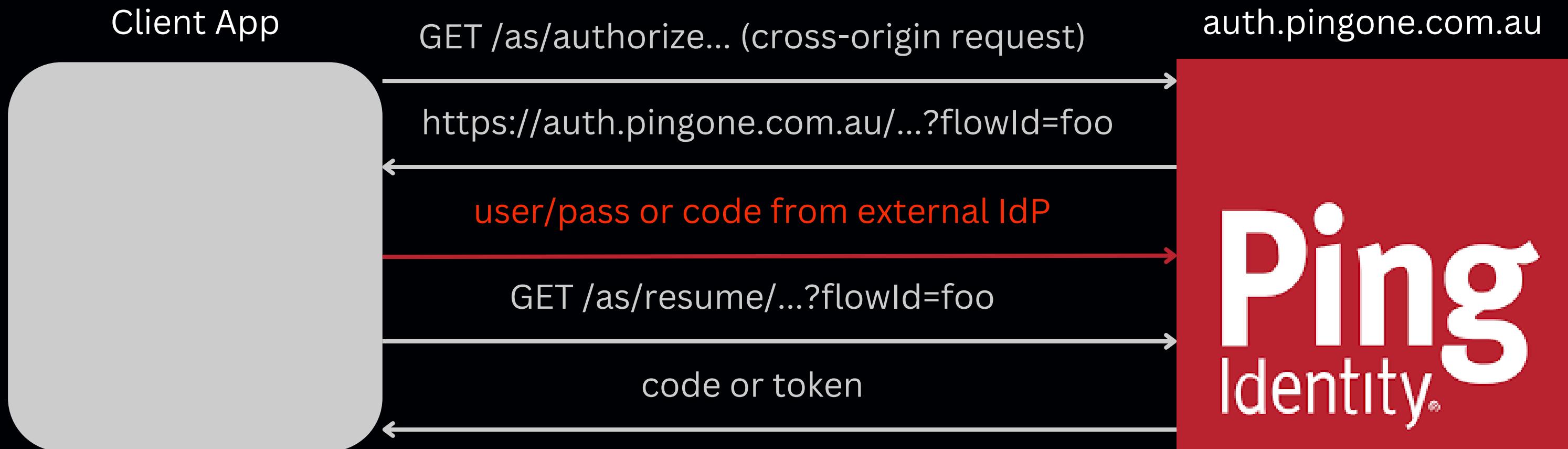
# Custom flows?



# Custom flows?



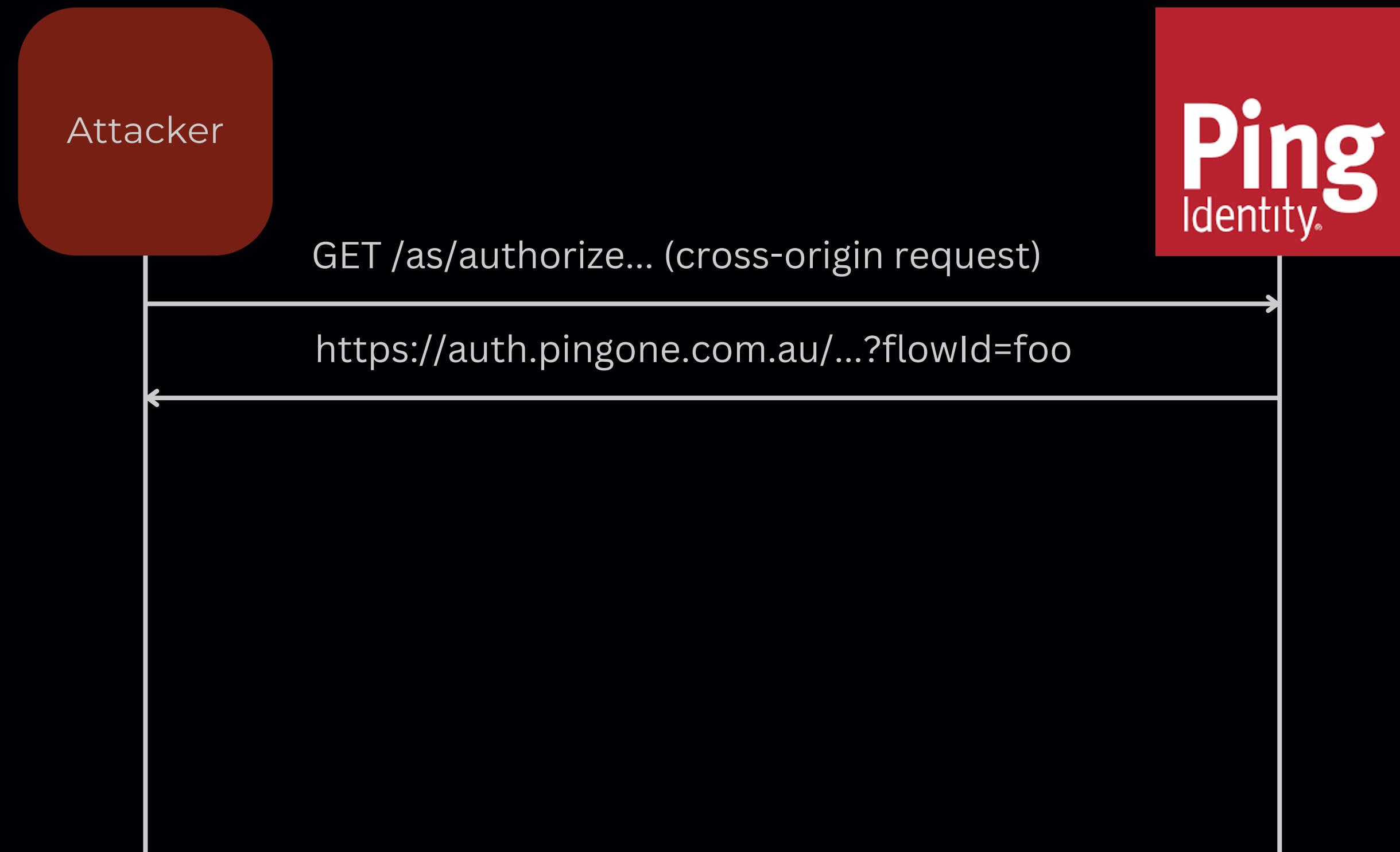
# Custom flows?



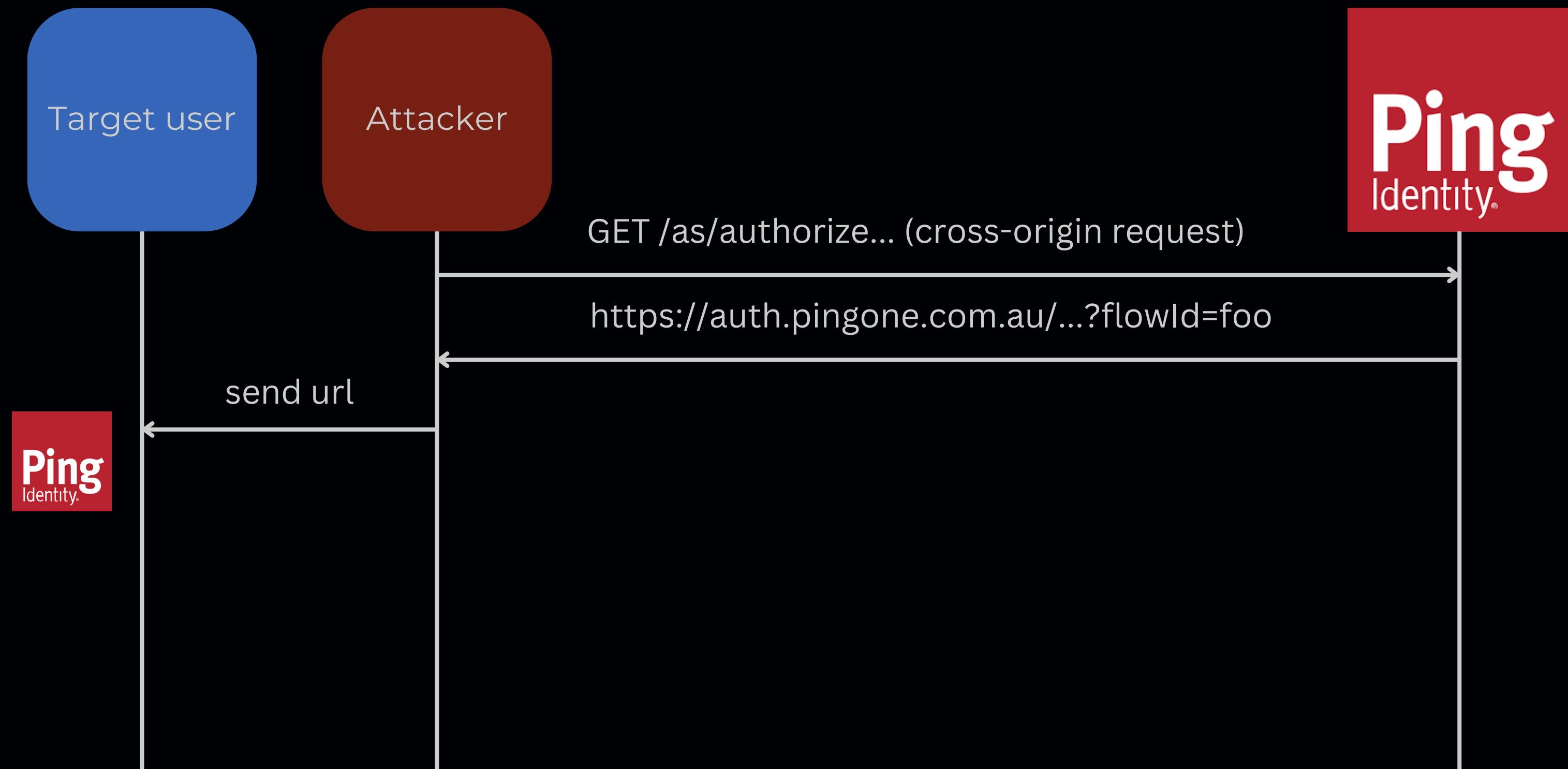
# Session fixation?

- Flow identifier acted as a session token
- Session fixation was a big issue in OAuth 1
- Attacker initiates a session then convinces target user into authenticating the session

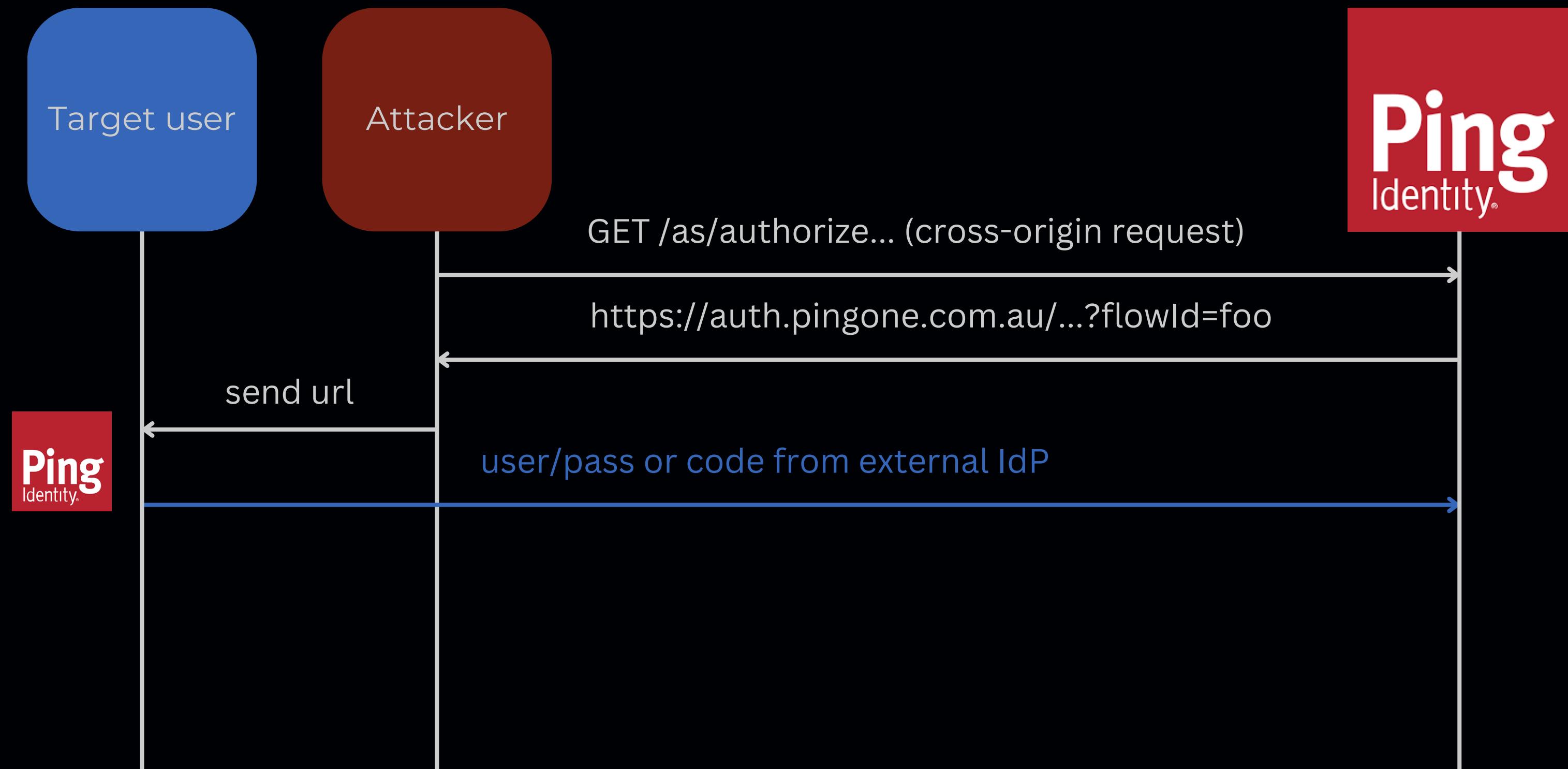
# Attack attempt 1



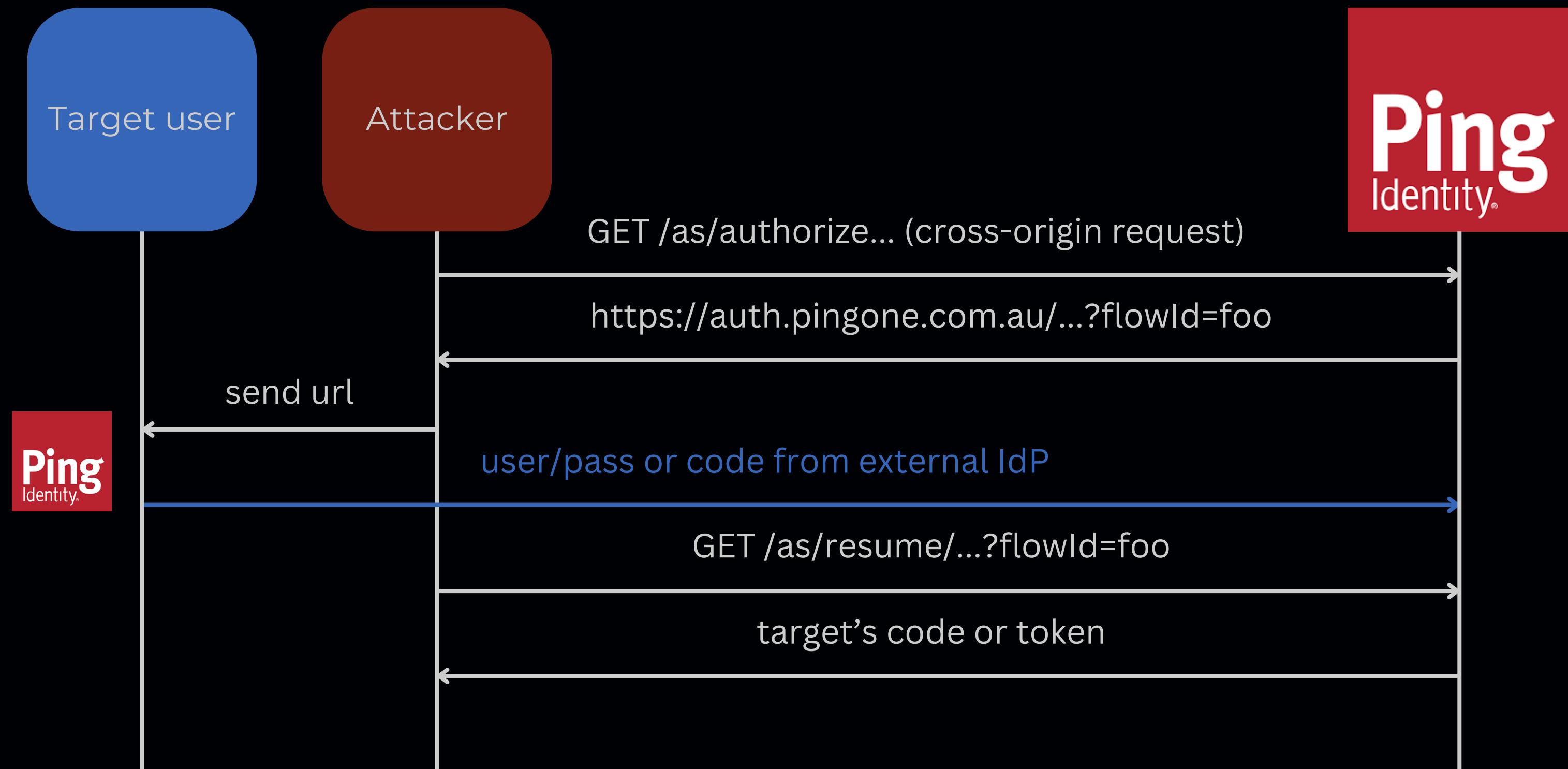
# Attack attempt 1



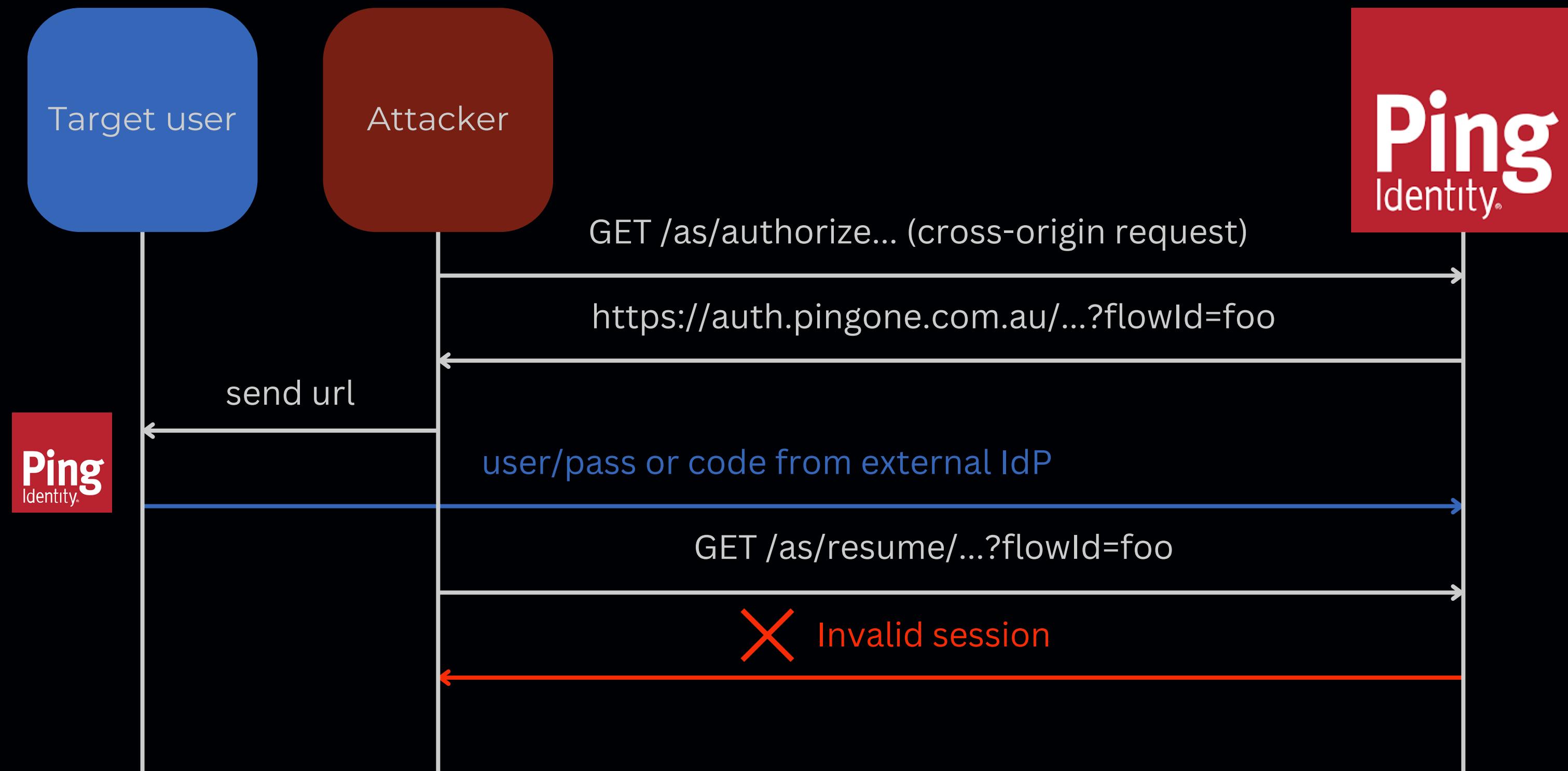
# Attack attempt 1



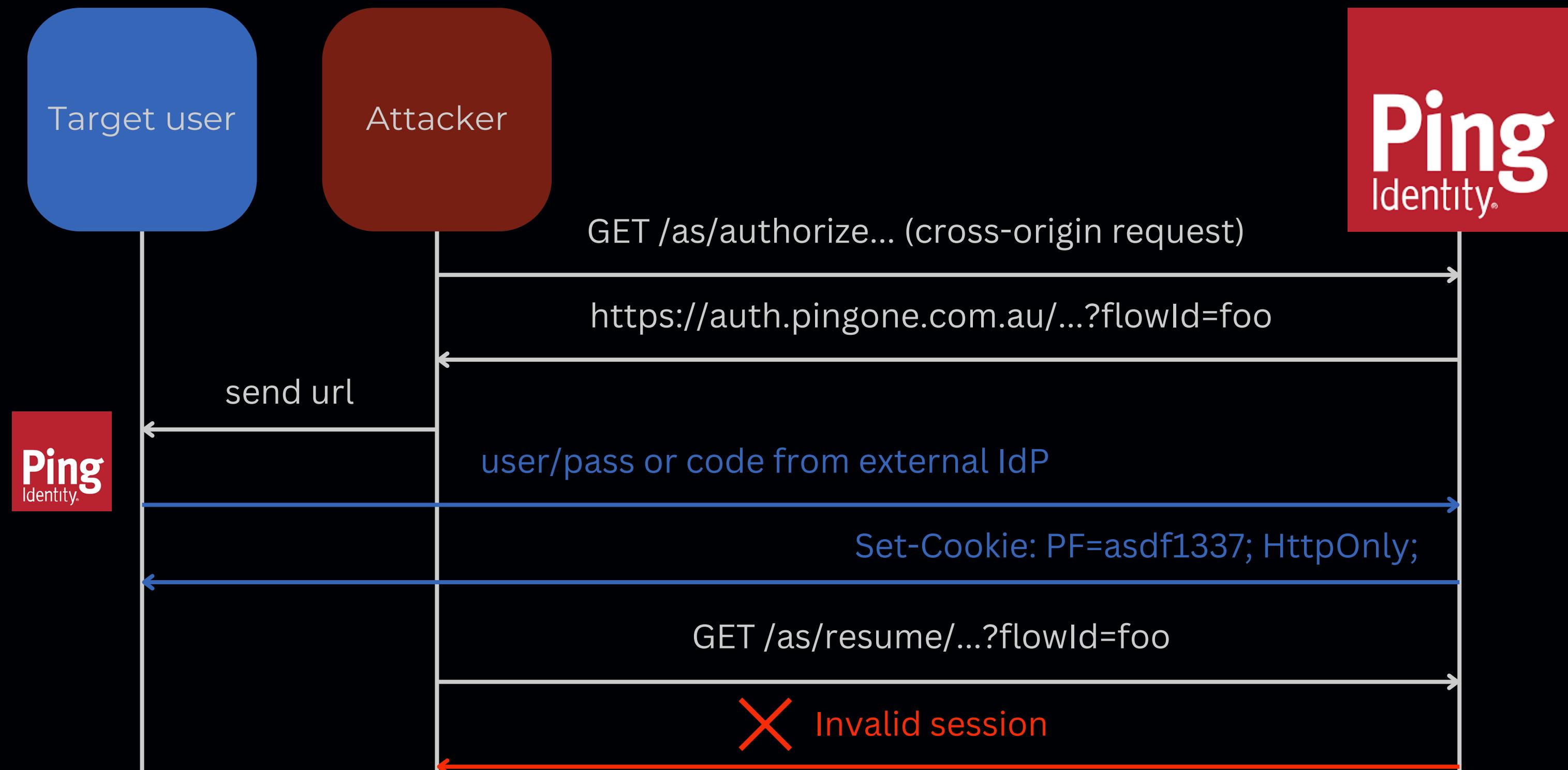
# Attack attempt 1



# Attack attempt 1



# Attack attempt 1



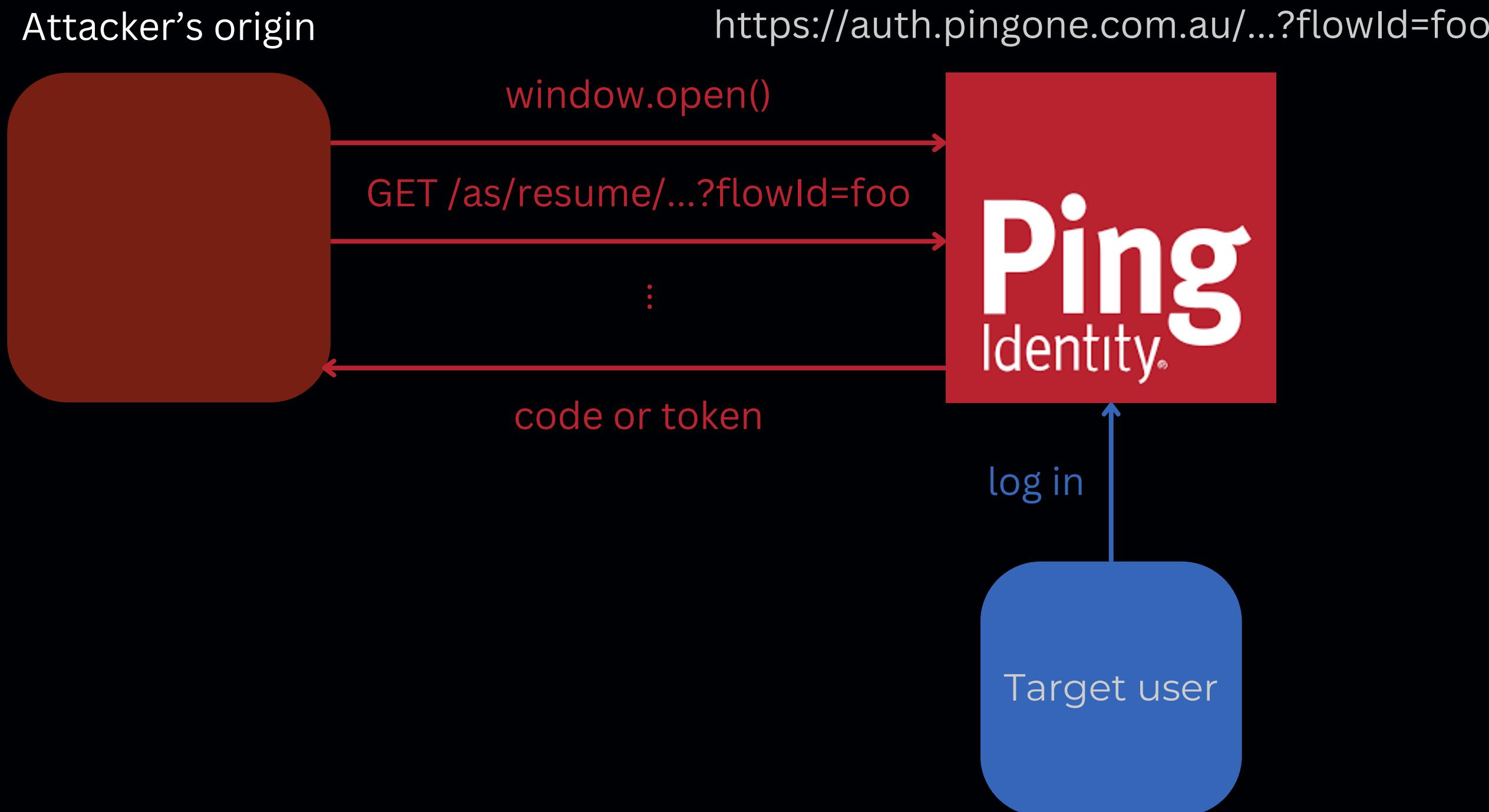
# Attack attempt 2



# Attack attempt 2



# Attack attempt 2



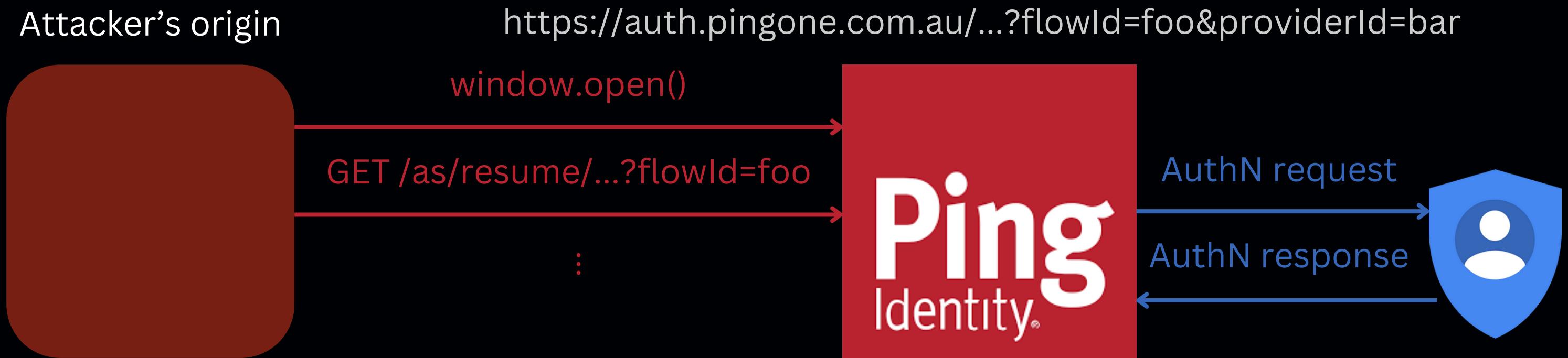
# Attack attempt 2



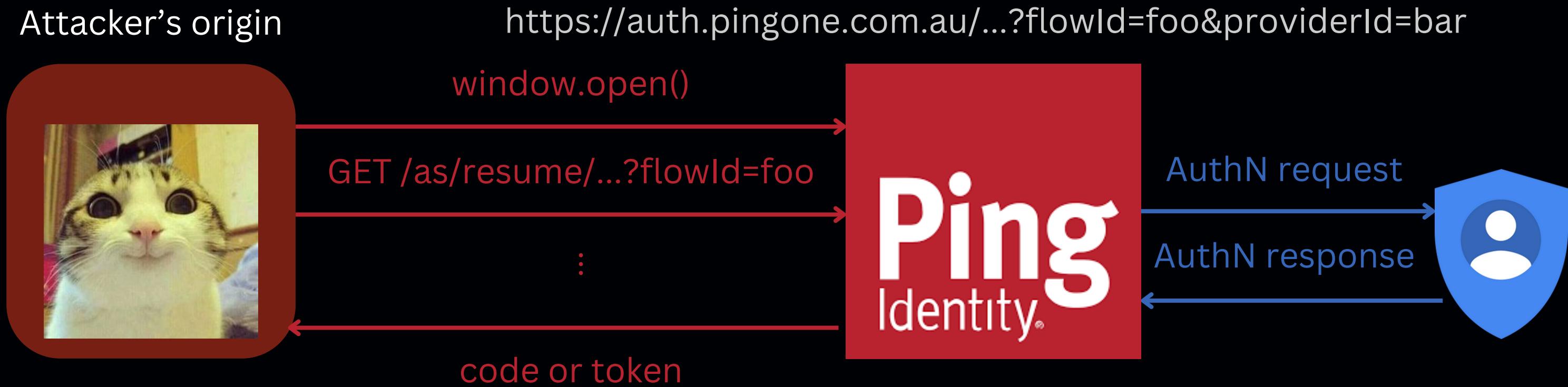
# Attack attempt 3



# Attack attempt 3



# Attack attempt 3



# RECAP

- pi.flow is a custom flow defined by Ping identity where code or token is returned within API response instead of redirect uri
- First attack (session fixation) didn't work because of a cookie set by pingone
- Created a website to perform attack with more steps
- Google SSO is your friend

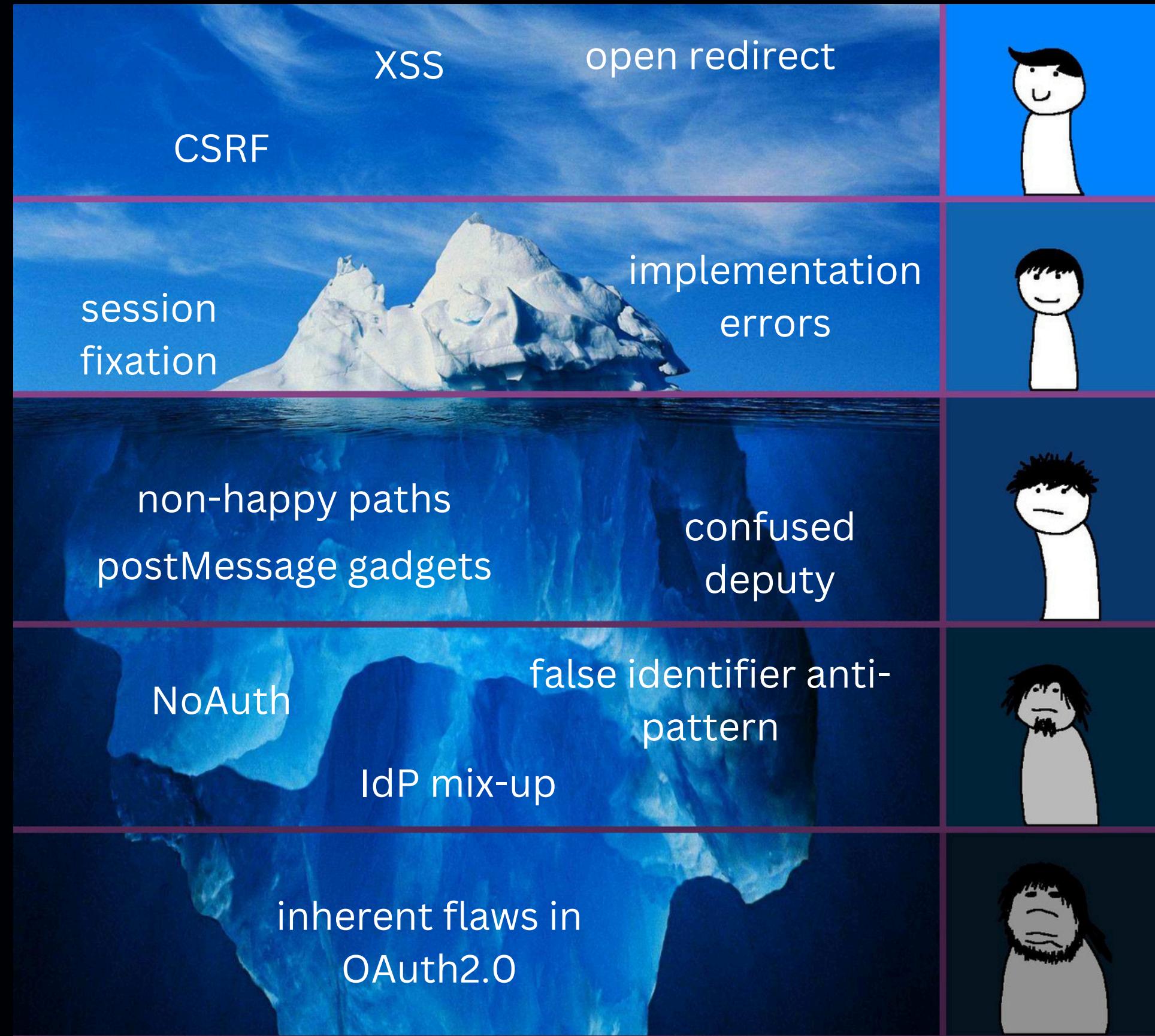
# What is up with that Ping flow

**i Note:** A browser-based application that interacts with the authorization endpoint using either `form_post` or `pi.flow` must protect itself against [Cross Site Scripting \(XSS\)](#) and [Cross Site Request Forgery \(CSRF\)](#) vulnerabilities. If a user has already authenticated using the browser, a malicious script running within the browser context that makes an authorization request using `form_post` or `pi.flow` may receive the user's access/id tokens directly in the response. To mitigate this risk, ensure your application follows these recommendations:

1. Disable support for the [OAuth 2.0 Implicit Grant Type](#).
2. Choose a Token Endpoint Authentication Method other than `NONE`.
3. Configure the application's `corsSettings.behavior` property to `ALLOW_SPECIFIC_ORIGINS` and in the `corsSettings.origins` property **only list your application's origins**. For more information, see [Application Operations](#).
4. Ensure that your application is written with [security best practices](#) to prevent script injection.

# WRAPUP

- Learnt something new about OIDC protocol
- Go and look for OAuth bugs (fun)



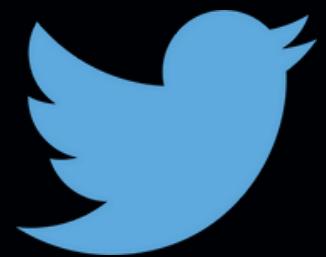
**THANKS FOR  
LISTENING**



# RESOURCES

- Frans Rosen's blog post: <https://labs.detectify.com/writeups/account-hijacking-using-dirty-dancing-in-sign-in-oauth-flows>
- Salt Security has done a LOT of OAuth research in the past year:  
<https://salt.security/blog/over-1-million-websites-are-at-risk-of-sensitive-information-leakage---xss-is-dead-long-live-xss>
- NoAuth Lore: <https://techcommunity.microsoft.com/t5/microsoft-entra-blog/the-false-identifier-anti-pattern/ba-p/3846013>
- Egor Homakov the GOAT: <https://homakov.blogspot.com/2014/02/how-i-hacked-github-again.html>
- Yep: <https://datatracker.ietf.org/doc/html/rfc6749#section-10.1>
- The Facebook 1click I mentioned: <https://ysamm.com/?p=763>
- For beginners: <https://portswigger.net/web-security/oauth#improper-implementation-of-the-implicit-grant-type>

# Contact me



@0xchsh



work: [chuanshu@sheasecurity.com.au](mailto:chuanshu@sheasecurity.com.au)  
personal: [jchuanshu@gmail.com](mailto:jchuanshu@gmail.com)

