

WebAuthn: How to get rid of password

Joost van Dijk, Yubico

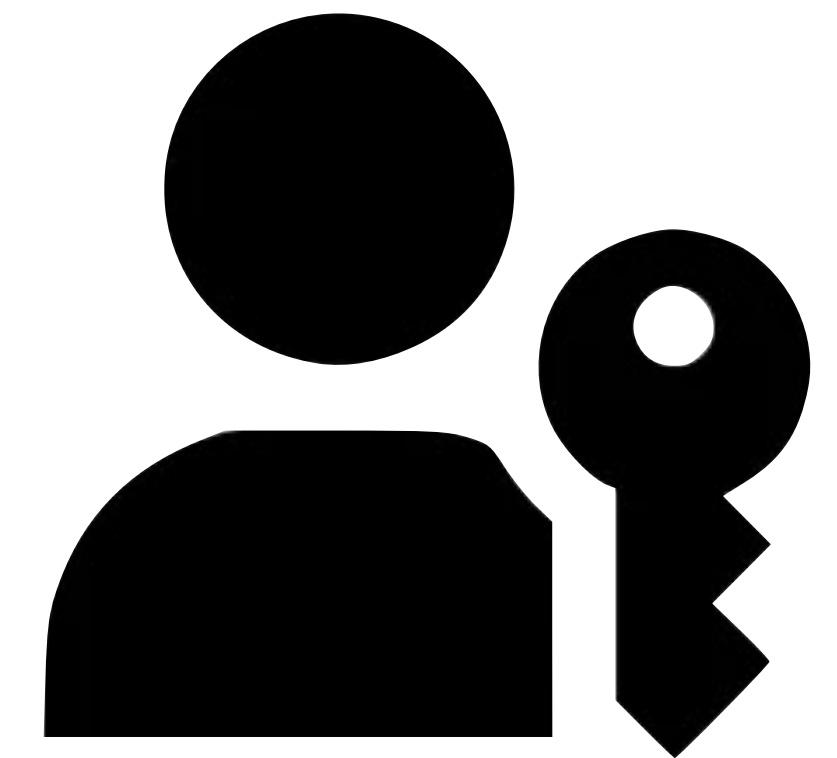
BSides Aarhus - 20 June 2026



yubico

What is a passkey?

- Passkeys are a more secure alternative to passwords
- More secure, because:
 - passkeys are resistant to phishing
 - passkeys have no secrets that can be leaked from servers
 - passkeys are generated automatically, never reused
 - passkeys can be stored on secure hardware: security keys
- Also easier to use:
 - No secrets to remember
 - *“Sign in with your face, your finger, or your PIN”*
 - Optionally, automatically backed up and synced





passkey.org



What is a passkey?

Passkeys are like passwords but better. They're better because they aren't created insecurely by humans and because they use public key cryptography to create much more secure experiences.

But passkeys aren't a new thing. It's just a new name starting to be used for WebAuthn/FIDO2 credentials that enable fully passwordless experiences. These types of credentials are also called discoverable credentials or sometimes resident credentials.

DEMO



Try it out

Create a temporary username so we can register your passkey. You'll only need your passkey to sign in after creating an account.

Sign up

OR

Sign in with passkey

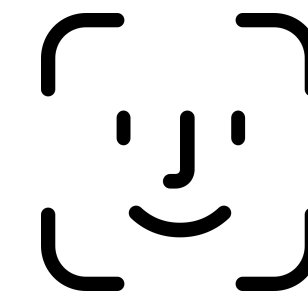
Accounts will be deleted after 24 hours. By registering, you agree that your data will be treated in accordance with the [Yubico Privacy Policy](#).

Passkeys are stored on Authenticators

- **Cross-Platform Authenticator**
also called *roaming* authenticator
example: a USB security key



- **Platform Authenticator**
Built into user's device
example: a built-in fingerprint sensor



FaceID



TouchID



Windows
Hello

- Roaming authenticators can use different *transports*: USB, NFC, BLE, hybrid

- Note: a single authenticator can store multiple passkeys!

Different types of passkeys

Device-bound



- Single-device
- Hardware attestation
- Ideal for high assurance use cases
- FIPS eligible
- Example: passkey stored on a security key

Synced | Copyable passkeys

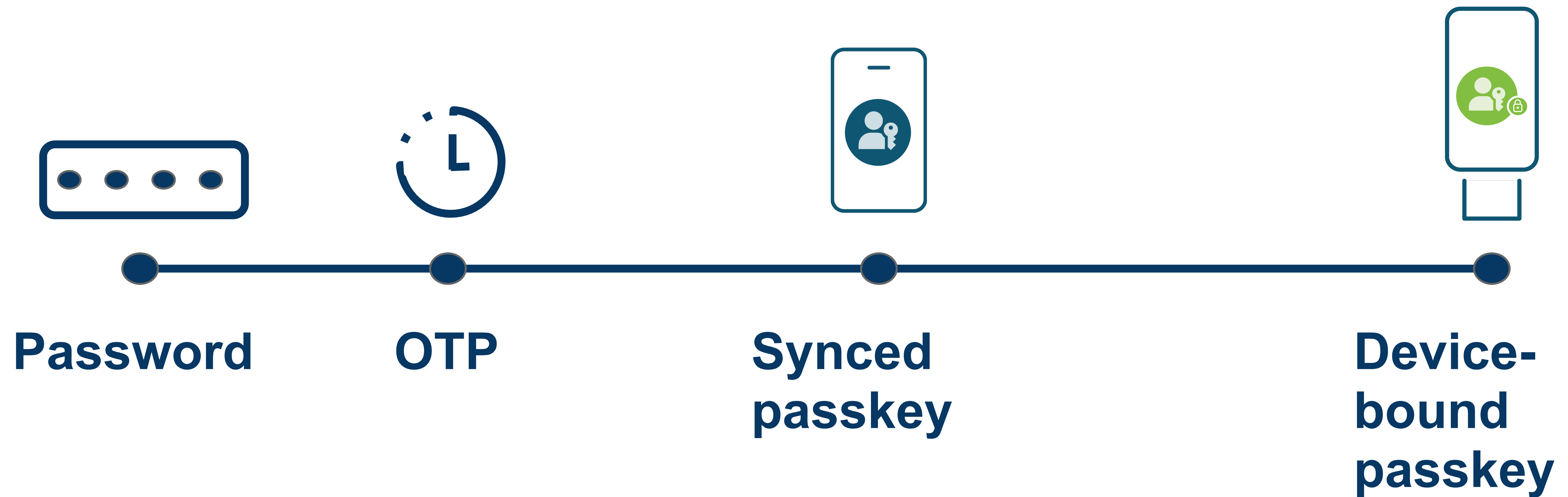


- Multi-device
- Backed up and synced across devices via a cloud provider
- No need to re-enroll a new device on every account!
- Synced across *devices* but not across *ecosystems* (i.e. a passkey synced via iCloud is not available on Android)

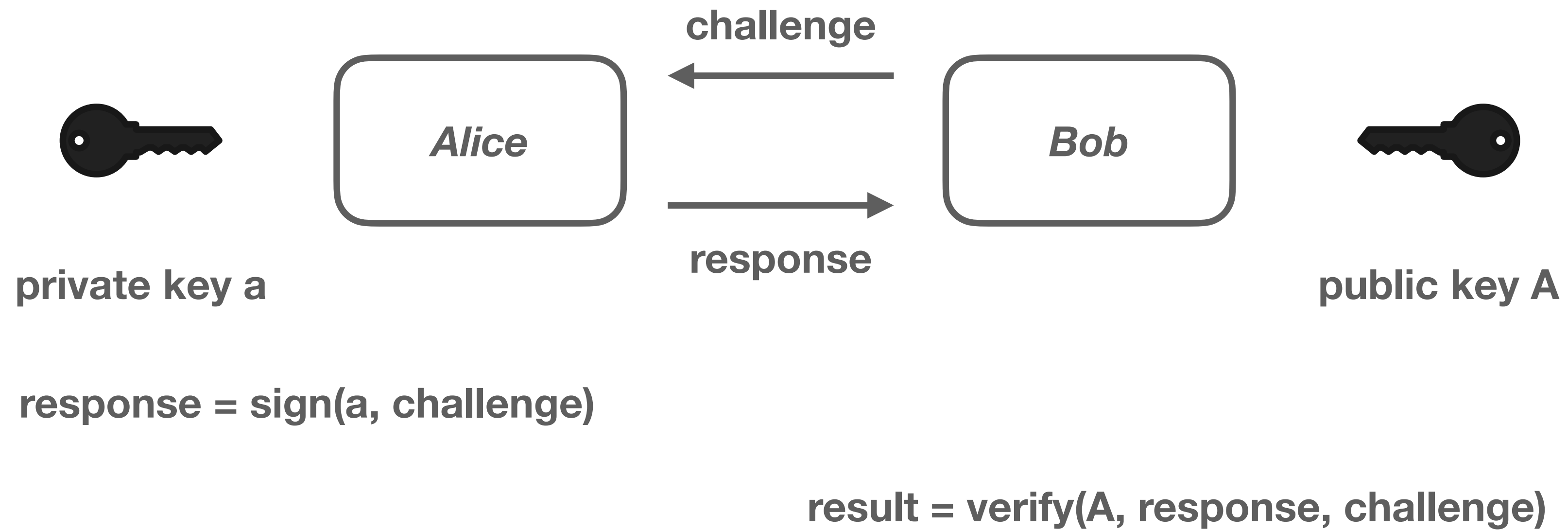
Assurance Levels

Low assurance

High assurance

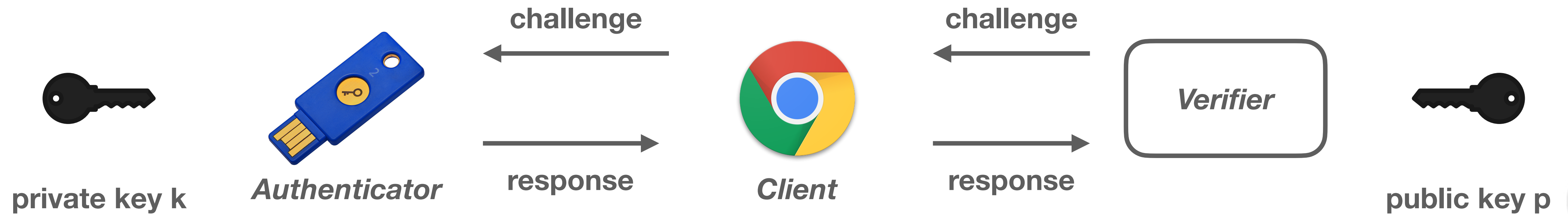


Public Key authentication



FIDO Public Key authentication

(Simplified)



$\text{response} = \text{sign}(k, \text{challenge})$

$\text{result} = \text{verify}(p, \text{response}, \text{challenge})$



https://login.live.youcantrustme.nl/login.srf?wa: ↻



Sign in

Email, phone or Skype

No account? [Create one!](#)

Next



Sign-in options

Phishing resistance

(Simplified)



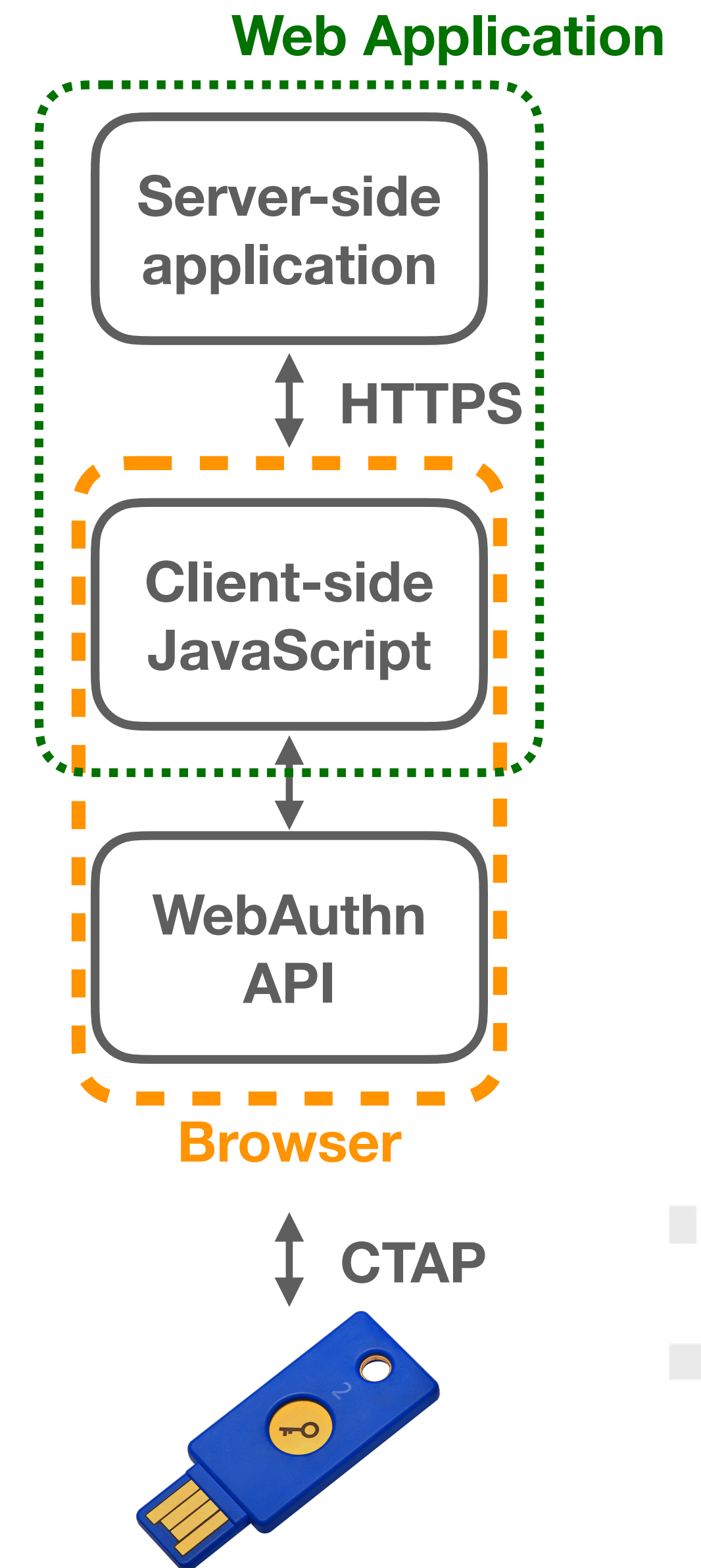
$\text{response} = \text{sign}(k, \text{challenge})$
+origin

$\text{result} = \text{verify}(p, \text{response}, \text{challenge})$
+origin

yubico

Webauthn: JavaScript API

- `navigator.credentials.create()`
register new FIDO credential
- `navigator.credentials.get()`
authenticate using a previously registered credential



WebAuthn API - authentication

- The `navigator.credentials.get()` operation initiates an authentication flow:
 - User selects an authenticator
 - User selects a credential (passkey)
 - User Presence/Verification
 - Authenticator signs the challenge with the matching private key
 - Returns assertion
- Assertion is relayed to server for verification

```
<script>
var getOptions = {
  publicKey: {
    challenge: serverChallenge
    // random nonce set by server
  }
}

function get() {
  navigator.credentials.get(getOptions)
    .then( (assertion) => {
      relayToServer(assertion)
    } )
}
</script>

<div>
  <button onClick="get()">get</button>
</div>
```

WebAuthn API - registration

- The `navigator.credentials.create()` operation initiates a registration flow:
 - User selects an authenticator
 - User Presence/Verification
 - Authenticator creates a key pair
 - and signs the challenge with the private key
 - Returns credential
- Credential is relayed to server for validation and storage

```
<script>
var createOptions = {
  publicKey: {
    rp: { name: "Example Relying Party" },
    user: { // set by server
      id: johnsUserID,
      name: "johnny",
      displayName: "John Doe"
    },
    pubKeyCredParams: [ ],
    authenticatorSelection: {
      residentKey: "required"
    },
    challenge: serverChallenge
  }
}
function create () {
  navigator.credentials.create(createOptions)
    .then( (credential) =>
    { relayToServer(credential) } )
}
</script>
<div>
  <button onClick="create()">create</button>
</div>
```

Backend API



Client app

JavaScript:
`fetch(optionsURL)`



Server app

```
POST /v1/assertion/options
Host: rp.example.com
Accept: application/json
Content-Type: application/json
{}
```

CTAP



JavaScript:
`credential.get(response)`

```
HTTP/1.1 200
Content-Type: application/json
{
  "requestId": "J4BJJeVl5y792xY-zyz8IAgyLiFYTdg-dBSQTA23hto",
  "publicKey": {
    "challenge": "2BxQbnxnaKrHFZB0KQF77j03M739v609VYBgEEYm4nU",
    "timeout": 180000,
    "rpId": "rp.example.com",
    "userVerification": "preferred"
  },
  "errorMessage": null
}
```

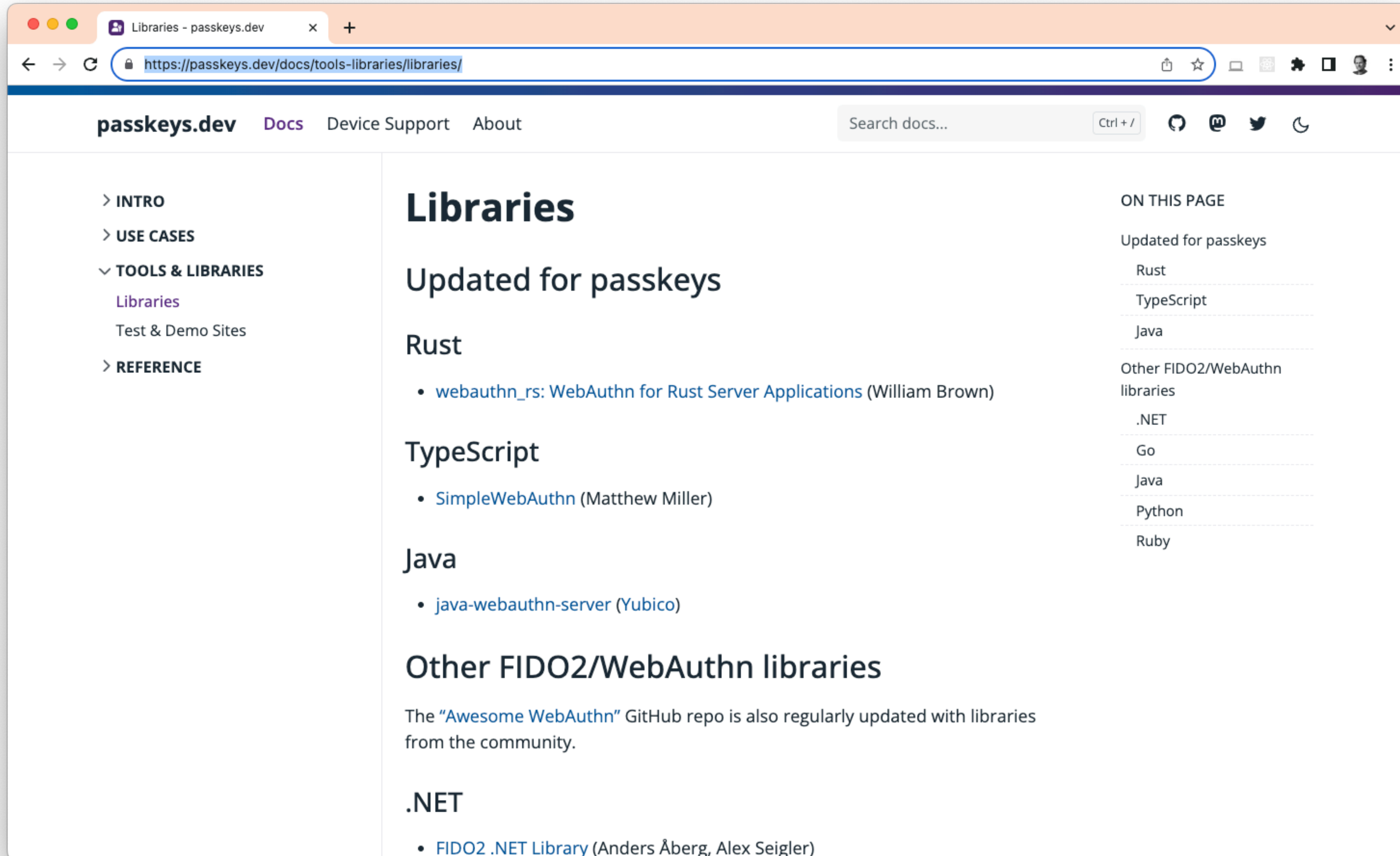
Yubico

Backend API



Passkey Libraries

<https://passkeys.dev/docs/tools-libraries/libraries/>



The screenshot shows a web browser window displaying the 'Libraries' page on the 'passkeys.dev' website. The browser's address bar shows the URL 'https://passkeys.dev/docs/tools-libraries/libraries/'. The website's navigation bar includes 'passkeys.dev', 'Docs', 'Device Support', and 'About', along with a search bar and social media icons. The main content area is divided into three columns. The left column is a sidebar with a tree view containing 'INTRO', 'USE CASES', 'TOOLS & LIBRARIES' (with 'Libraries' selected), 'Test & Demo Sites', and 'REFERENCE'. The middle column features the main heading 'Libraries' and a sub-heading 'Updated for passkeys'. Below this, there are sections for 'Rust' (with a link to 'webauthn_rs: WebAuthn for Rust Server Applications'), 'TypeScript' (with a link to 'SimpleWebAuthn'), 'Java' (with a link to 'java-webauthn-server'), and 'Other FIDO2/WebAuthn libraries' (with a paragraph about the 'Awesome WebAuthn' GitHub repo). The right column, titled 'ON THIS PAGE', lists 'Updated for passkeys' followed by links for 'Rust', 'TypeScript', 'Java', and 'Other FIDO2/WebAuthn libraries', which are further broken down into links for '.NET', 'Go', 'Java', 'Python', and 'Ruby'.

passkeys.dev Docs Device Support About

Search docs... Ctrl + /

> INTRO

> USE CASES

▼ TOOLS & LIBRARIES

Libraries

Test & Demo Sites

> REFERENCE

Libraries

Updated for passkeys

Rust

- [webauthn_rs: WebAuthn for Rust Server Applications](#) (William Brown)

TypeScript

- [SimpleWebAuthn](#) (Matthew Miller)

Java

- [java-webauthn-server](#) (Yubico)

Other FIDO2/WebAuthn libraries

The "Awesome WebAuthn" GitHub repo is also regularly updated with libraries from the community.

.NET

- [FIDO2 .NET Library](#) (Anders Åberg, Alex Seigler)

ON THIS PAGE

Updated for passkeys

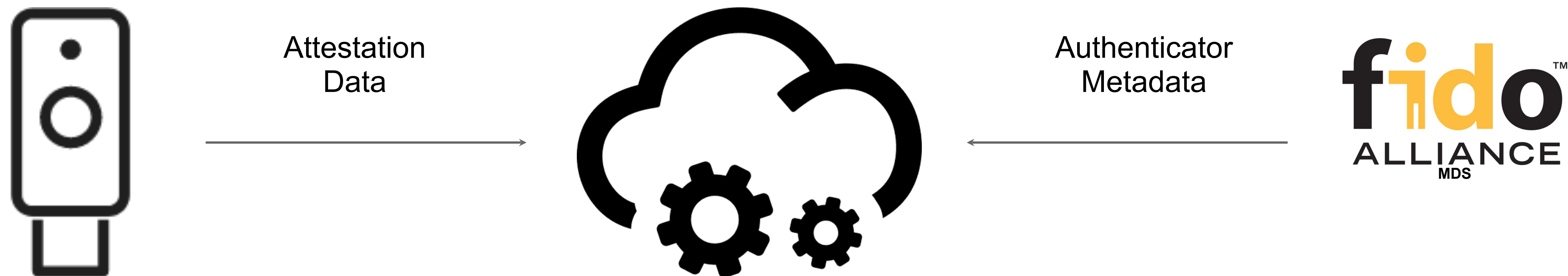
- Rust
- TypeScript
- Java

Other FIDO2/WebAuthn libraries

- .NET
- Go
- Java
- Python
- Ruby

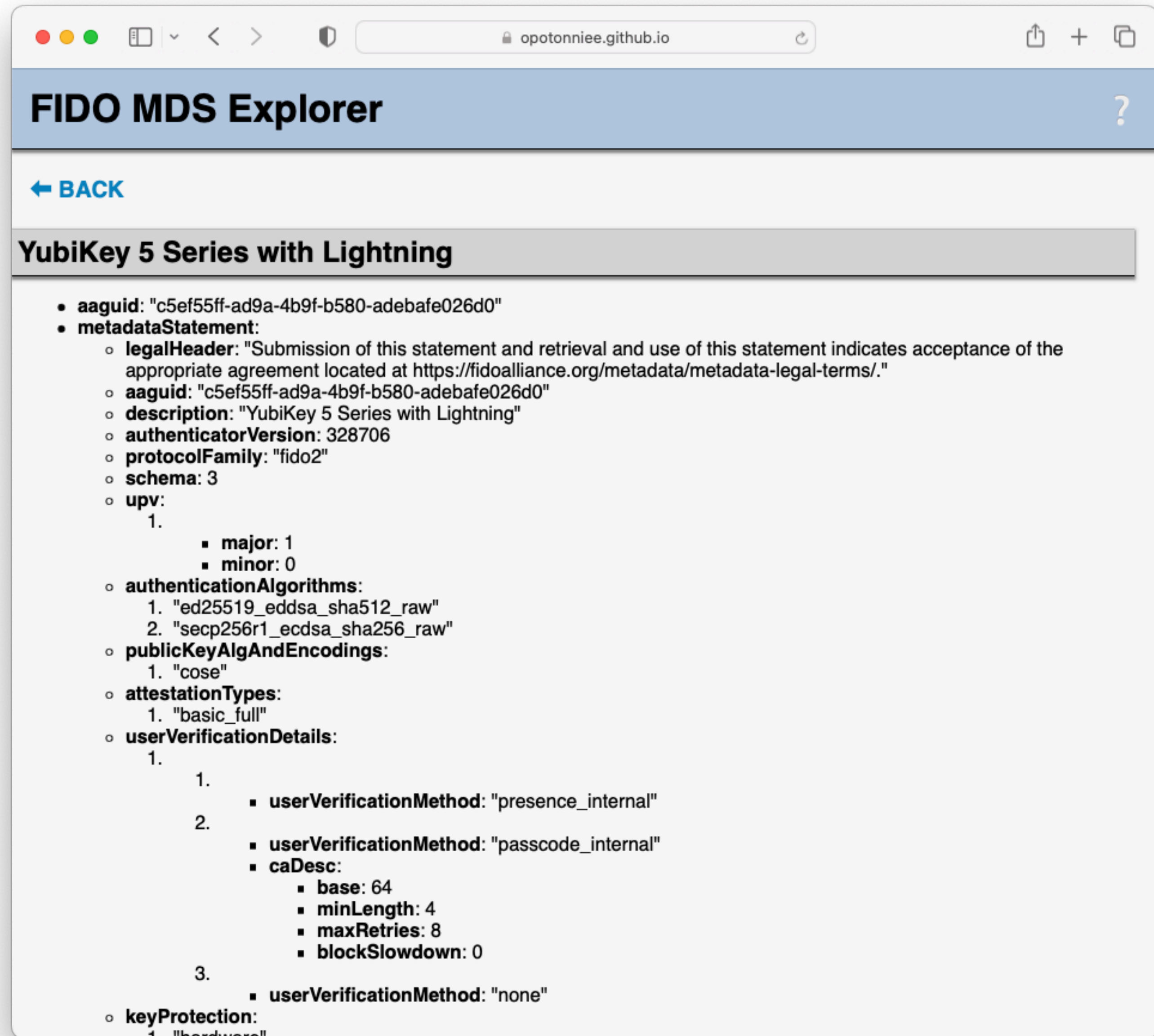
Attestation and Metadata

- Attestation provides verifiable evidence as to the authenticator's origin
- Based on a hardware attestation key and certificate
- Use FIDO Alliance Metadata Service to determine provenance
- Implement Allow/Deny lists to filter Authenticators
- Typically used in high-assurance (enterprise) use cases



Metadata example

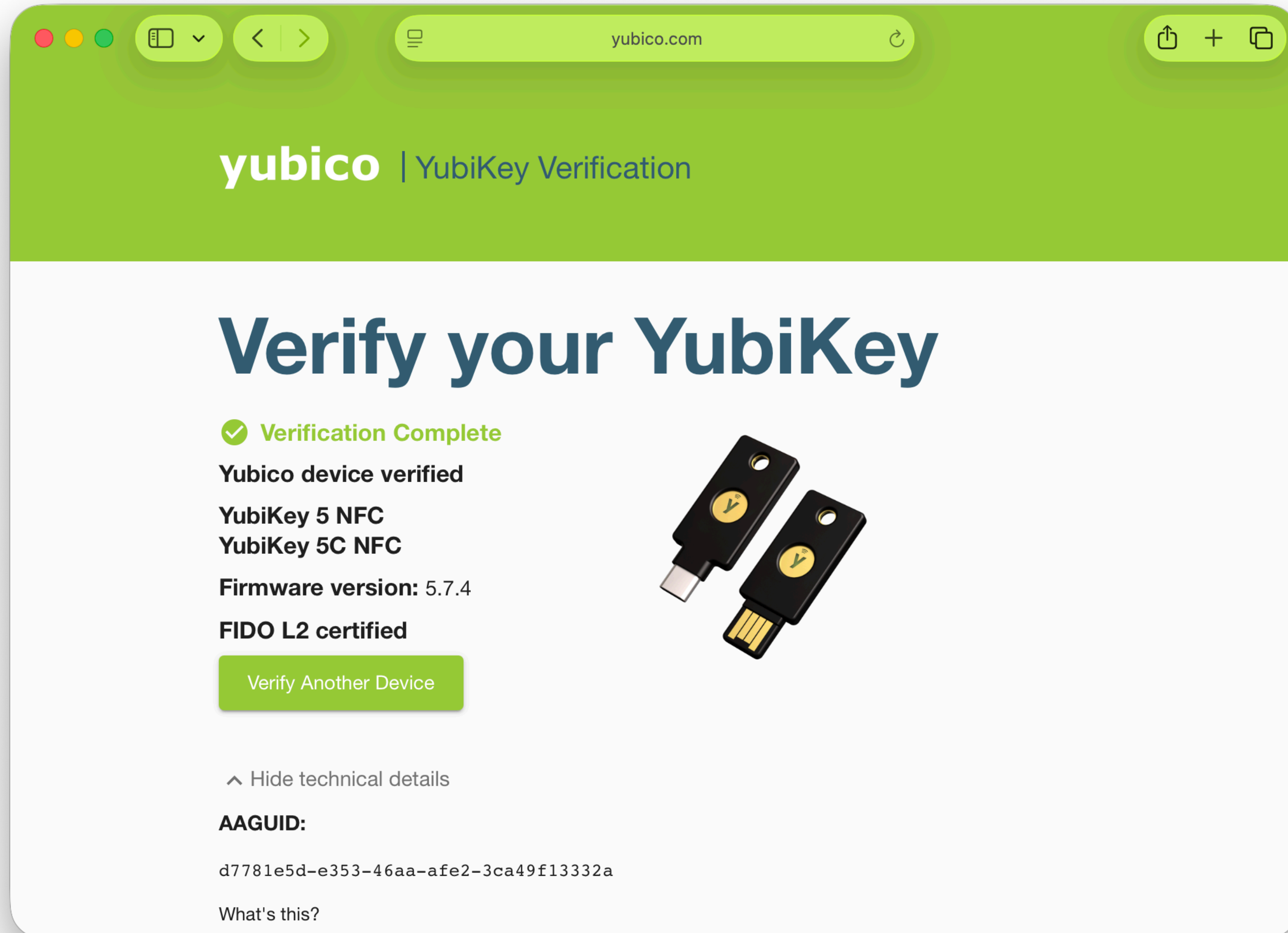
- `aaguid`
(Authenticator unique ID)
- `keyProtection`
e.g. `secure_element`
- `transports`
e.g. `usb`
- `status`
(certification level)



The screenshot shows a web browser window with the address bar containing `opotonniee.github.io`. The page title is "FIDO MDS Explorer". Below the title is a "BACK" button. The main content area displays the metadata for a "YubiKey 5 Series with Lightning". The metadata is structured as follows:

- `aaguid`: "c5ef55ff-ad9a-4b9f-b580-adebaf026d0"
- `metadataStatement`:
 - `legalHeader`: "Submission of this statement and retrieval and use of this statement indicates acceptance of the appropriate agreement located at <https://fidoalliance.org/metadata/metadata-legal-terms/>."
 - `aaguid`: "c5ef55ff-ad9a-4b9f-b580-adebaf026d0"
 - `description`: "YubiKey 5 Series with Lightning"
 - `authenticatorVersion`: 328706
 - `protocolFamily`: "fido2"
 - `schema`: 3
 - `upv`:
 1.
 - `major`: 1
 - `minor`: 0
 - `authenticationAlgorithms`:
 1. "ed25519_eddsa_sha512_raw"
 2. "secp256r1_ecdsa_sha256_raw"
 - `publicKeyAlgAndEncodings`:
 1. "cose"
 - `attestationTypes`:
 1. "basic_full"
 - `userVerificationDetails`:
 1.
 - `userVerificationMethod`: "presence_internal"
 2.
 - `userVerificationMethod`: "passcode_internal"
 - `caDesc`:
 - `base`: 64
 - `minLength`: 4
 - `maxRetries`: 8
 - `blockSlowdown`: 0
 3.
 - `userVerificationMethod`: "none"
 - `keyProtection`:
 1. "hardware"

<https://www.yubico.com/genuine/>



yubico | YubiKey Verification

Verify your YubiKey

✔ **Verification Complete**

Yubico device verified

YubiKey 5 NFC
YubiKey 5C NFC

Firmware version: 5.7.4

FIDO L2 certified


Verify Another Device

^ Hide technical details

AAGUID:

d7781e5d-e353-46aa-afe2-3ca49f13332a

[What's this?](#)



yubico

<https://demo.yubico.com/webauthn-developers>

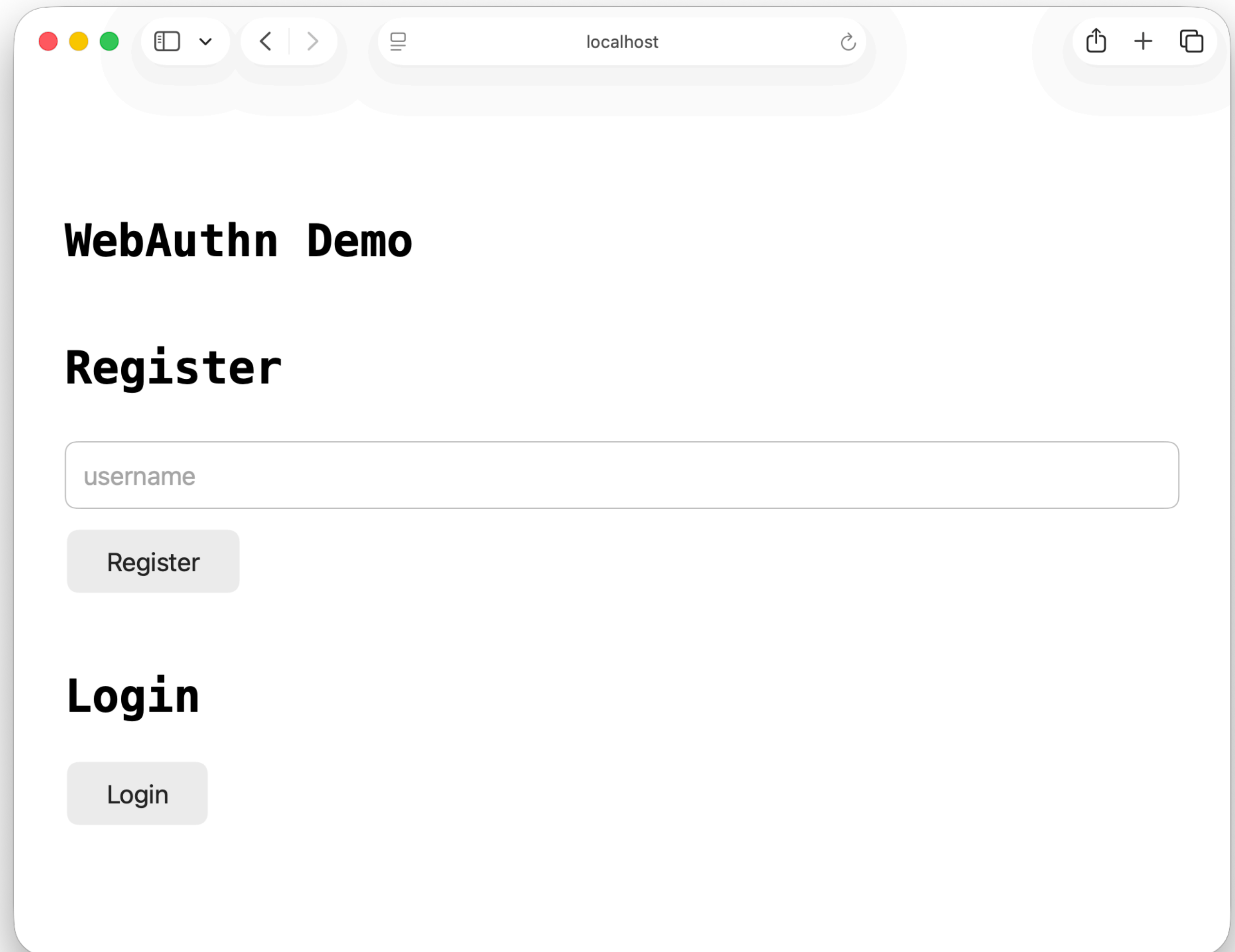
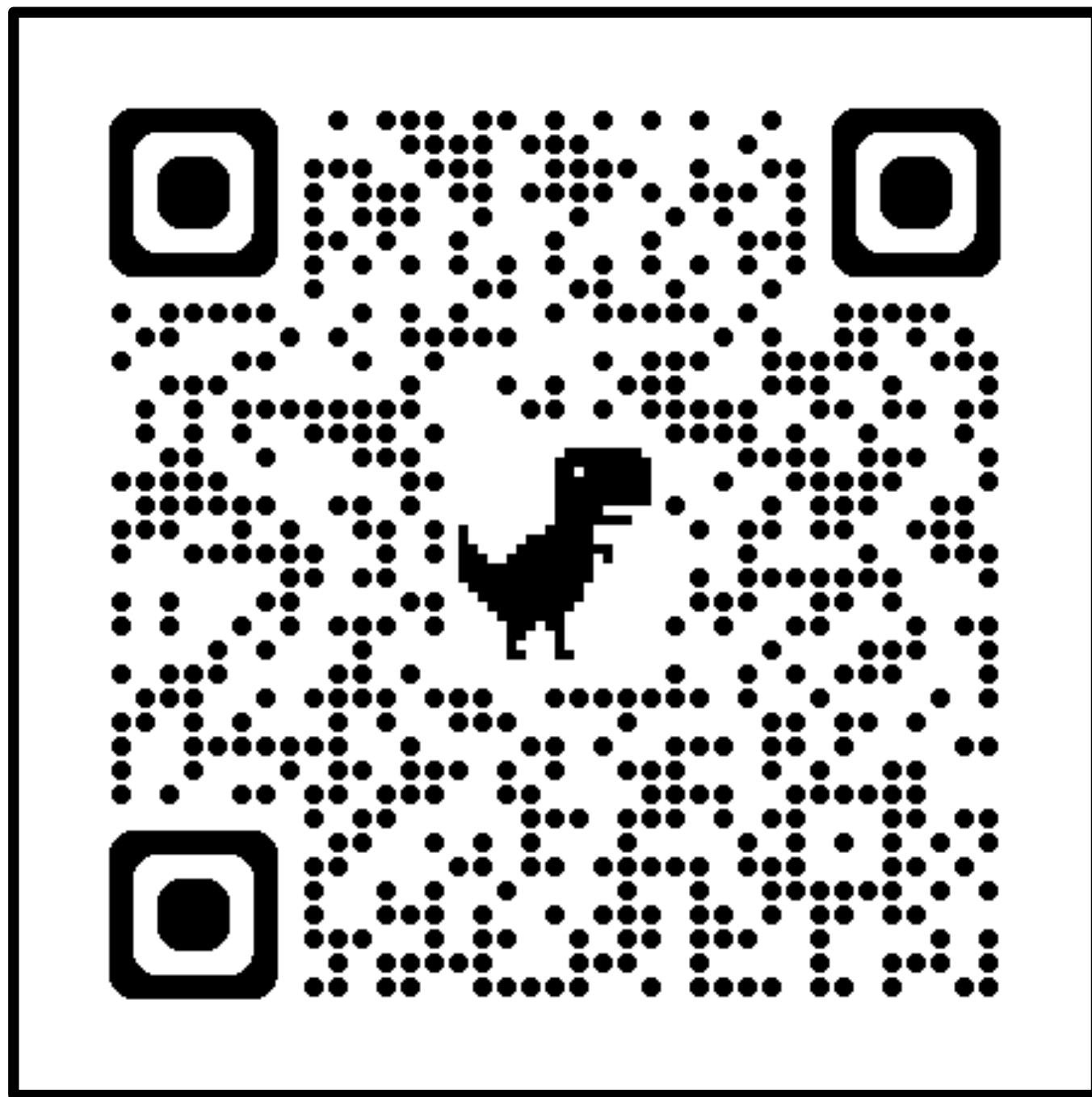
The screenshot shows the 'CREATE' tab of the Yubico webauthn-developers demo. The interface includes a 'RESET' button, a 'Preview' toggle, and several input fields for user identity: 'user' (set to 'Defined by session'), 'user.id' (6311424384b464124c4), 'user.name' (Ms3qfgCvbr8W), and 'user.displayName' (Ms3qfgCvbr8W). There are also sections for 'Authenticator selection' and 'Other options'. A 'Preview' window is open, displaying a JSON object:

```
{
  "publicKey": {
    "challenge": "3db099ecaebf3cc6cd53cd83f2f54d00",
    "rp": {
      "id": "demo.yubico.com",
      "name": "Yubico Demo"
    },
    "user": {
      "id": "6311424384b464124c4df659cf417f9d83e3e09c71b2651",
      "name": "Ms3qfgCvbr8W",
      "displayName": "Ms3qfgCvbr8W"
    },
    "attestation": "direct",
    "authenticatorSelection": {
      "userVerification": "preferred",
      "residentKey": "discouraged",
      "requireResidentKey": false
    },
    "excludeCredentials": [],
    "timeout": 90000,
    "extensions": {
      "credProps": true
    }
  }
}
```

Below the JSON is a text input field with the placeholder 'Or paste something to decode...' and a close button. The right sidebar contains 'Settings' (Binary format: hex, b64, b64u, js), 'Session Management' (Session: Ms3qfgCvbr8W, NEW, RESTORE), and 'Credentials' (No Credentials, Create some credentials to use to authenticate a session).

Do It Yourself

Backend: demo.py
Frontend: static/index.html



<https://tinyurl.com/bsides-passkeys>

Android Passkey Credential Provider Service

- Release Date:
Mon 22 June 2026

The screenshot shows the Google Play Store page for the 'YubiKey Passkey Enabler' app by Yubico AB. The app is categorized under 'Apps' and has a rating of 5+ downloads. The main title is 'YubiKey Passkey Enabler' in a large, bold font. Below the title, the developer's name 'Yubico AB' is displayed. There are two buttons: 'Install on more devices' and 'Share'. Below these buttons, there are two informational messages: 'This app is available for some of your devices' and 'You can share this with your family. Learn more about Family Library.' The app's icon is a green square with a white person silhouette and a key with a 'Y' on it. Below the main content, there are four preview cards showing the app's interface on a smartphone. The first card is titled 'Set up your passkey' and shows a 'Passkey service not enabled' error message. The second card is titled 'Discover your NFC reader' and shows a hand tapping a YubiKey on the back of the phone. The third card is titled 'Secure your identity' and shows a 'Passkey service enabled' confirmation message. The fourth card is titled 'Enable passkeys on Android' and shows the 'Passkeys, passkeys & accounts' settings screen. To the right of the preview cards, there is a PEGI 3 rating box with the text 'PEGI 3 Learn more'. Below the PEGI rating, there is an 'App support' dropdown menu and a 'More apps to try' section featuring the Spotify app.

Google Play Games Apps Books Kids

YubiKey Passkey Enabler

Yubico AB

5+ Downloads

Install on more devices Share

This app is available for some of your devices You can share this with your family. [Learn more about Family Library](#)

Set up your passkey
In settings, configure your phone to be able to use passkeys over USB or NFC.

Discover your NFC reader
Easily see where to tap your YubiKey® based on the NFC reader's position in your Android phone.

Secure your identity
Use your YubiKey®, containing the strongest hardware-backed passkeys, to securely log in across websites and apps that support passkeys.

Enable passkeys on Android
Plug in your YubiKey®, a physical security key that delivers hardware-backed passkeys for a phishing-resistant method to log in.

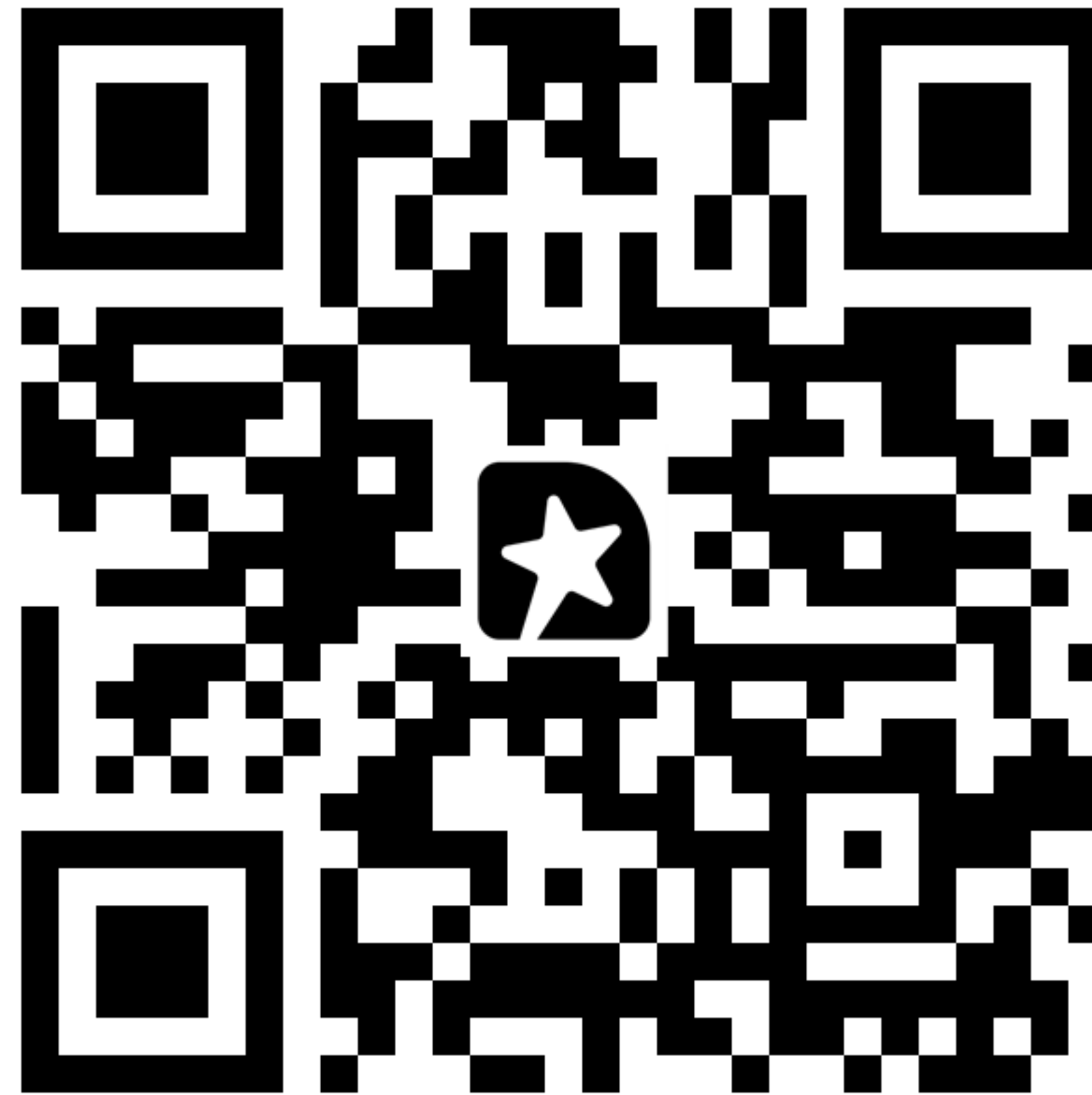
3 PEGI 3
[Learn more](#)

App support ▾

More apps to try →

Spotify: Music and Podcasts
Spotify AB
4.3 ★

Questions?



WebAuthn: How to get rid of passwords.