WebAuthn:
How to get rid of password

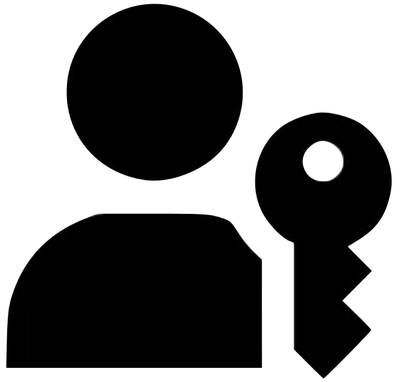
Joost van Dijk, Yubico

BSides Aarhus - 20 June 2026

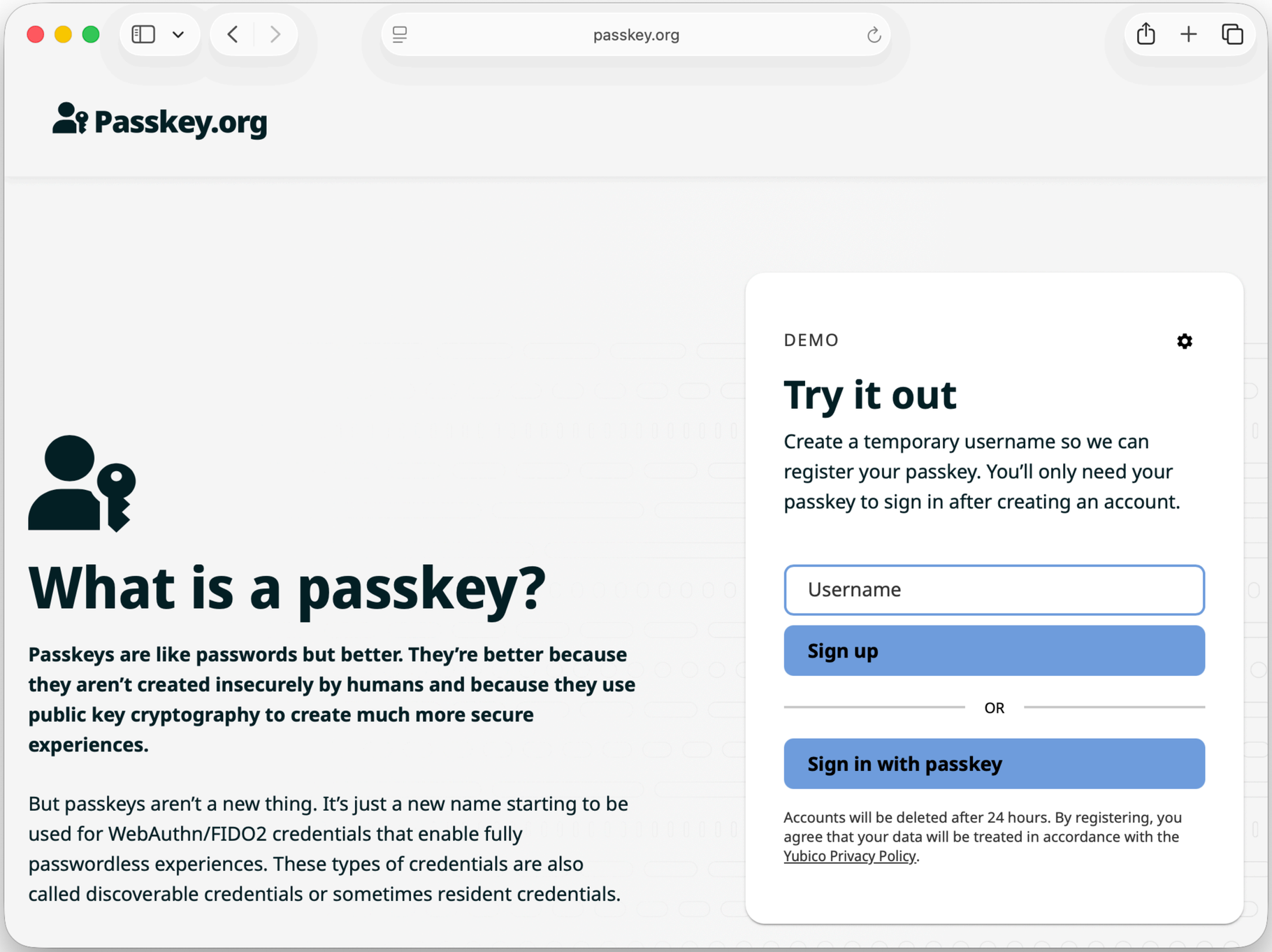
yubico

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



yubico



yubico

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)

yubico

Assurance Levels

Low assurance

High assurance

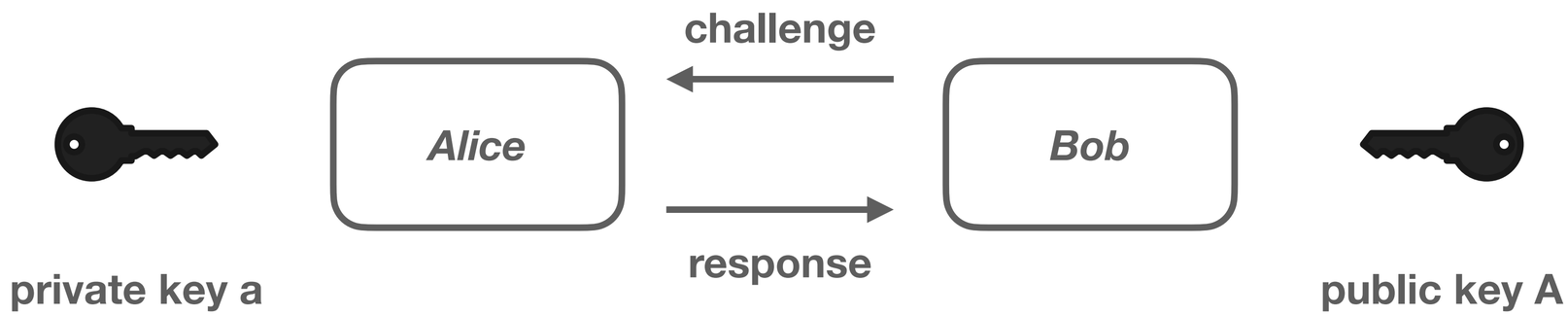
Password

OTP

Synced passkey

Device-bound passkey

Public Key authentication



response = sign(a, challenge)

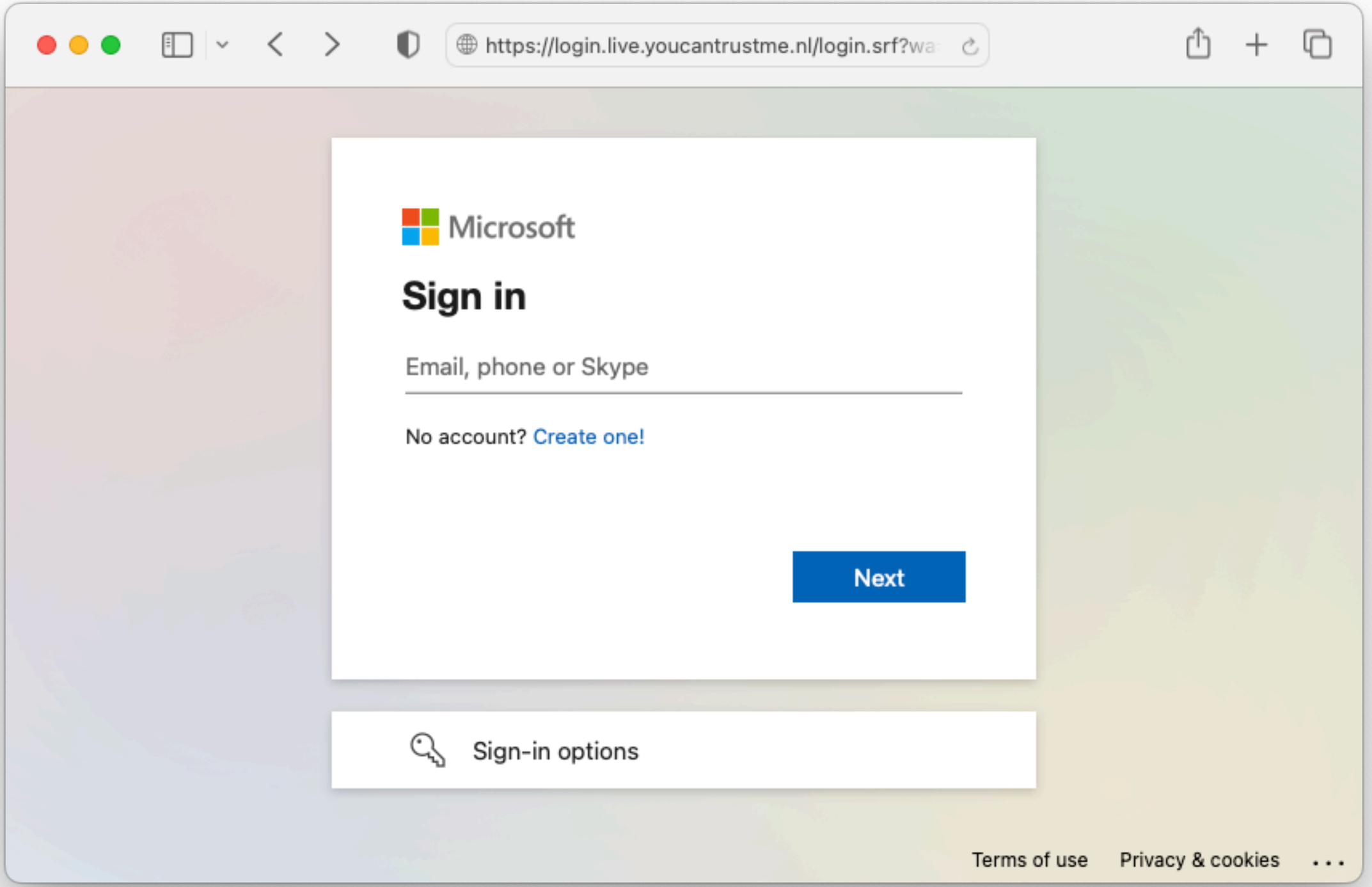
result = verify(A, response, challenge)

FIDO Public Key authentication

(Simplified)

response = sign(k, challenge)

result = verify(p, response, challenge)


https://login.live.youcantrustme.nl/login.srf?wa
Microsoft
Sign in
Email, phone or Skype
No account? Create one!
Next
Sign-in options
Terms of use
Privacy & cookies

Phishing resistance

(Simplified)

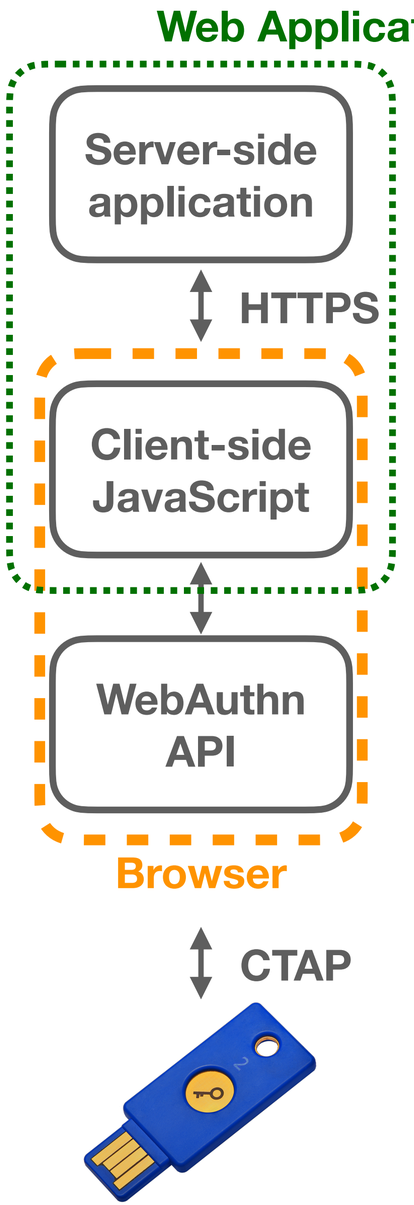
response = sign(k, challenge)
+origin

result = verify(p, response, challenge)
+origin

yubico

Webauthn: JavaScript API

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



yubico

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

Authenticator

Client app

Relying Party

Server app

JavaScript:
fetch(optionsURL)

```
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": "2BxQbnxnaKrHFZB0KQF77j03M739v6O9VYBgEEYm4nU",
    "timeout": 180000,
    "rpId": “rp.example.com”,
    "userVerification": "preferred"
  },
  "errorMessage": null
}
```

Backend API

Authenticator

CTAP

Client app

JavaScript:
fetch(optionsURL)

Relying Party

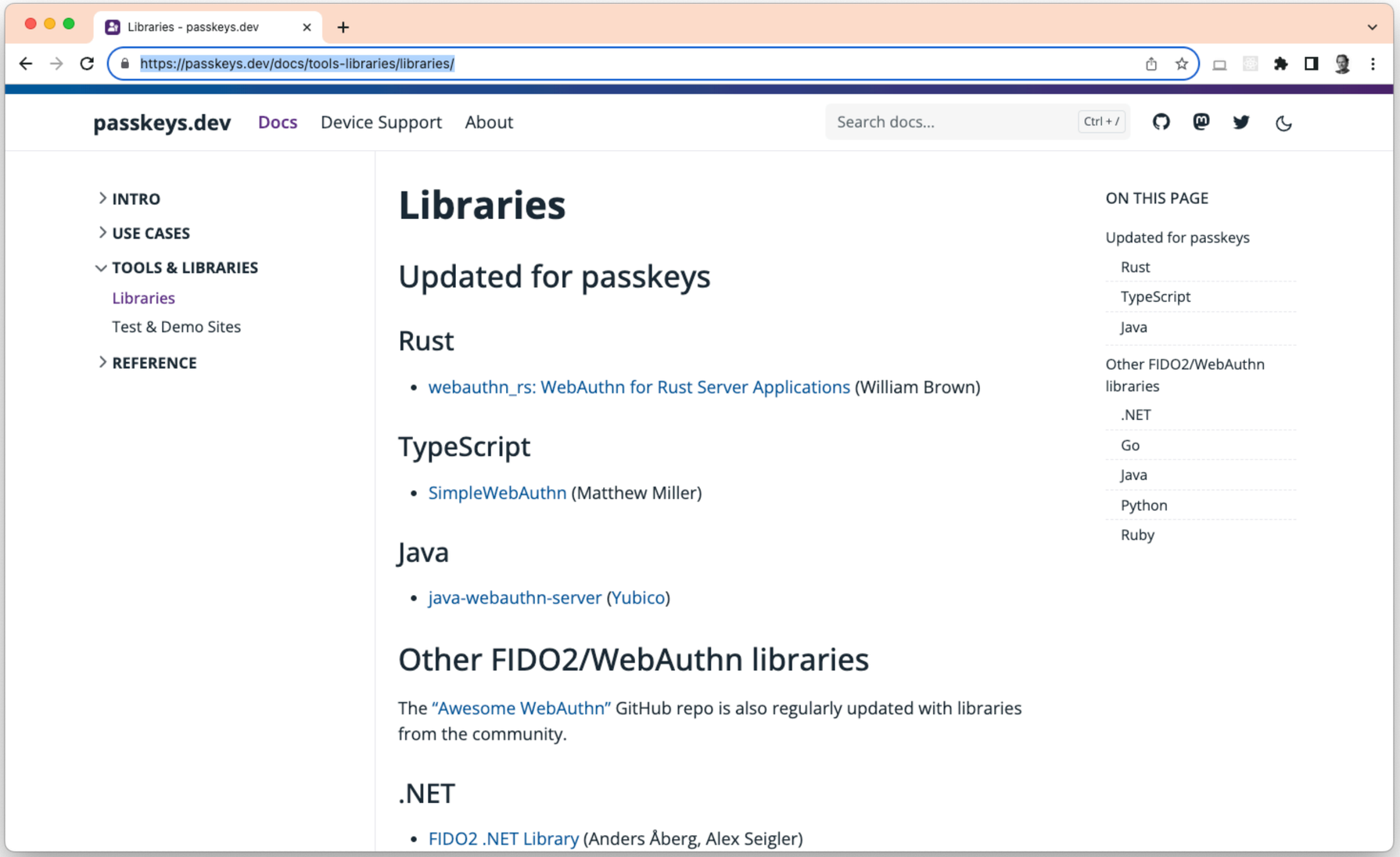
Server app

```
POST /v1/assertion/result

Host: rp.example.com
Accept: application/json
Content-Type: application/json

{
  "requestId": "J4BJJeVl5y792xY-zyz8IAgyLiFYTdg-dBSQTA23hto",
  "assertionResult": {
    "id": "LFdoCFJTyB82ZzSJUHc-c72yraRc_1mPvG",
    "response": {
      "authenticatorData": "SZYN5Gh0NBcPZHZgW4_krrmihjzzuoMdl2MBAAAAAA...",
      "signature": "ME8fLjd5y6TUOLWt5l9DQIhANiYig9newAJZYTzG1i5lwP",
      "userHandle": "jdoe",
      "clientDataJSON": "eyJjaGFTBrTmM4uIjoiaHR0cDovL2xvY2FsF1dGhuLmdldCJ9..."
    },
    "type": "public-key"
 }
}
```

```
HTTP/1.1 200
Content-Type: application/json

{
  "status": "ok"
}
```

Passkey Libraries

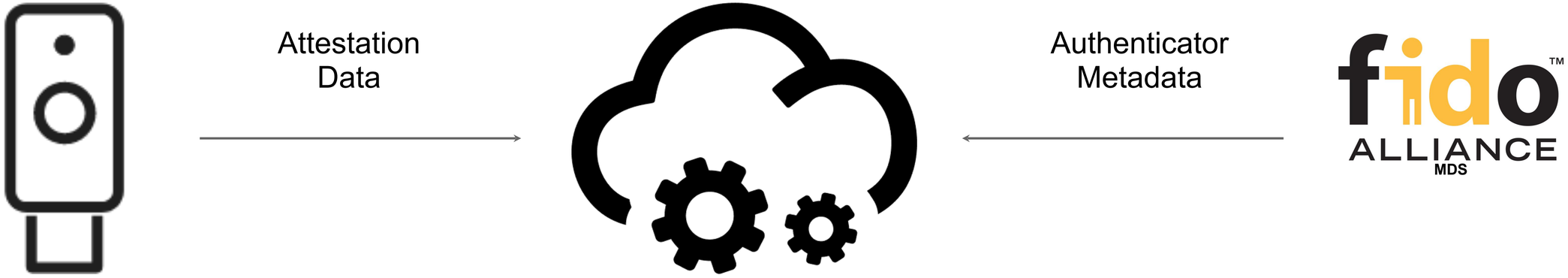
https://passkeys.dev/docs/tools-libraries/libraries/



yubico

Attestation and Metadata

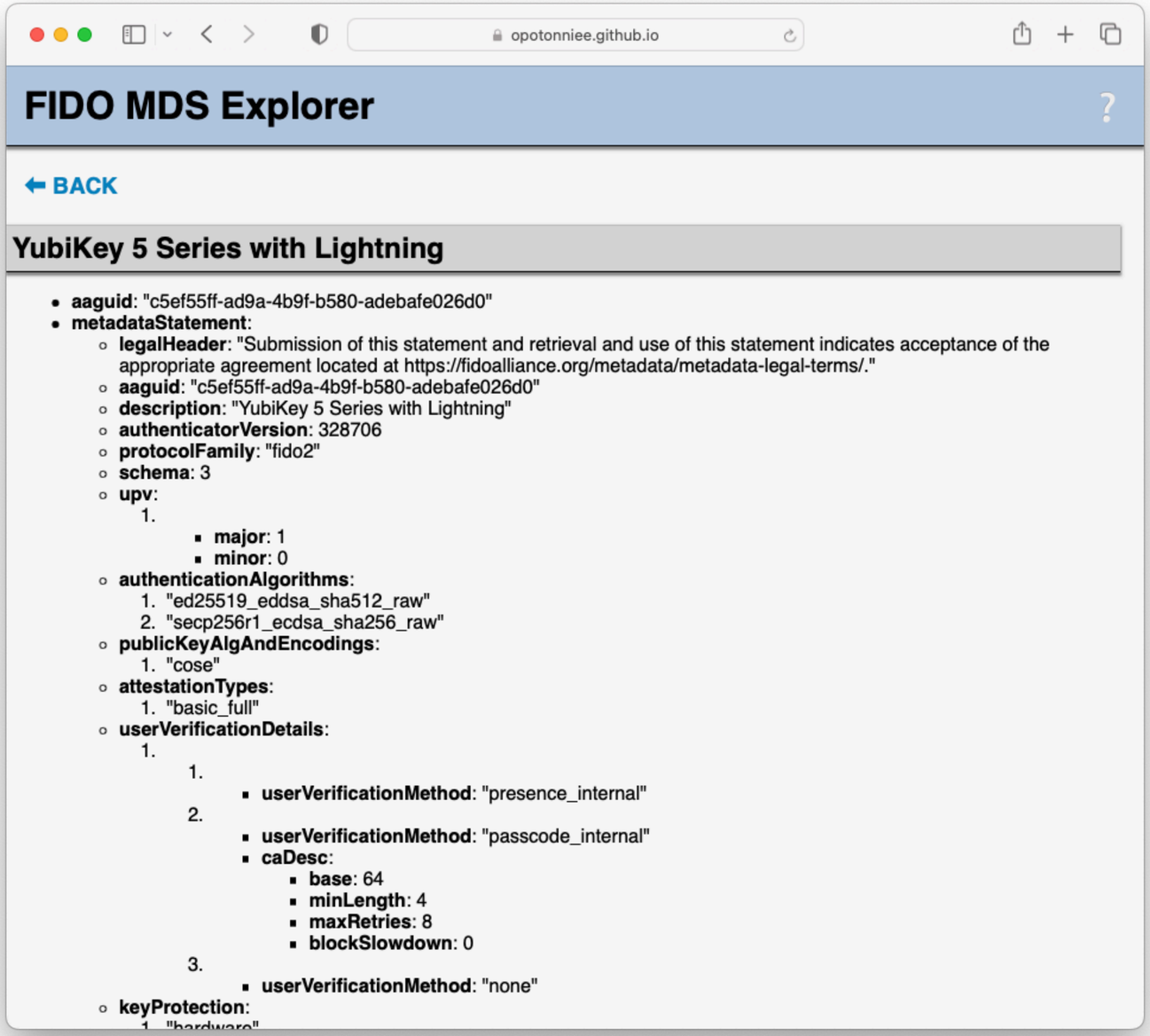
- Attestation provides verifiable evidence as to the authenticator’s origin
- Based on a hardware attestation key and certificate
- Use FIDO Alliance Metadata Service to determine provenance
- Implement Allow/Deny lists to filter Authenticators
- Typically used in high-assurance (enterprise) use cases



yubico

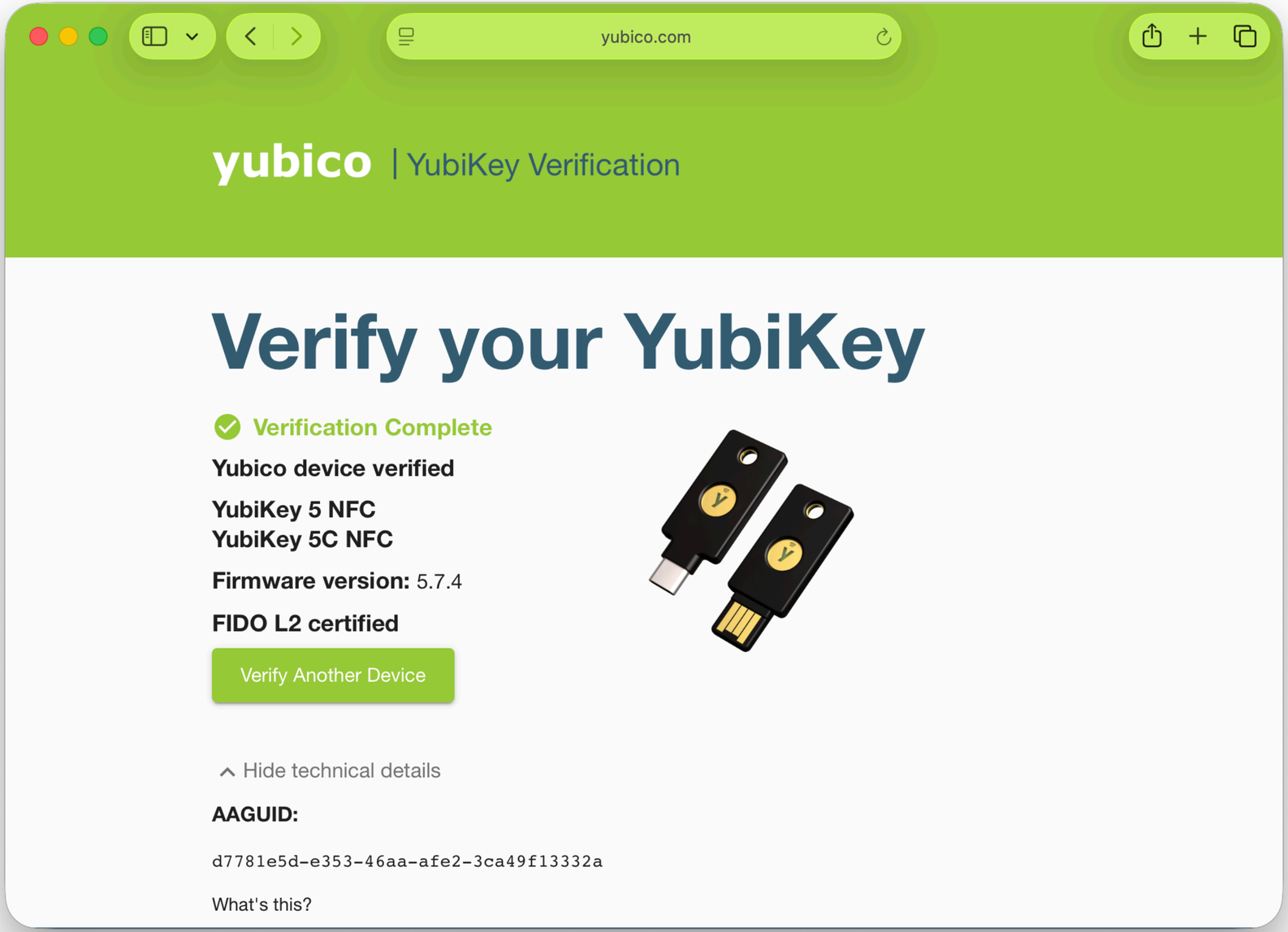
Metadata example

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



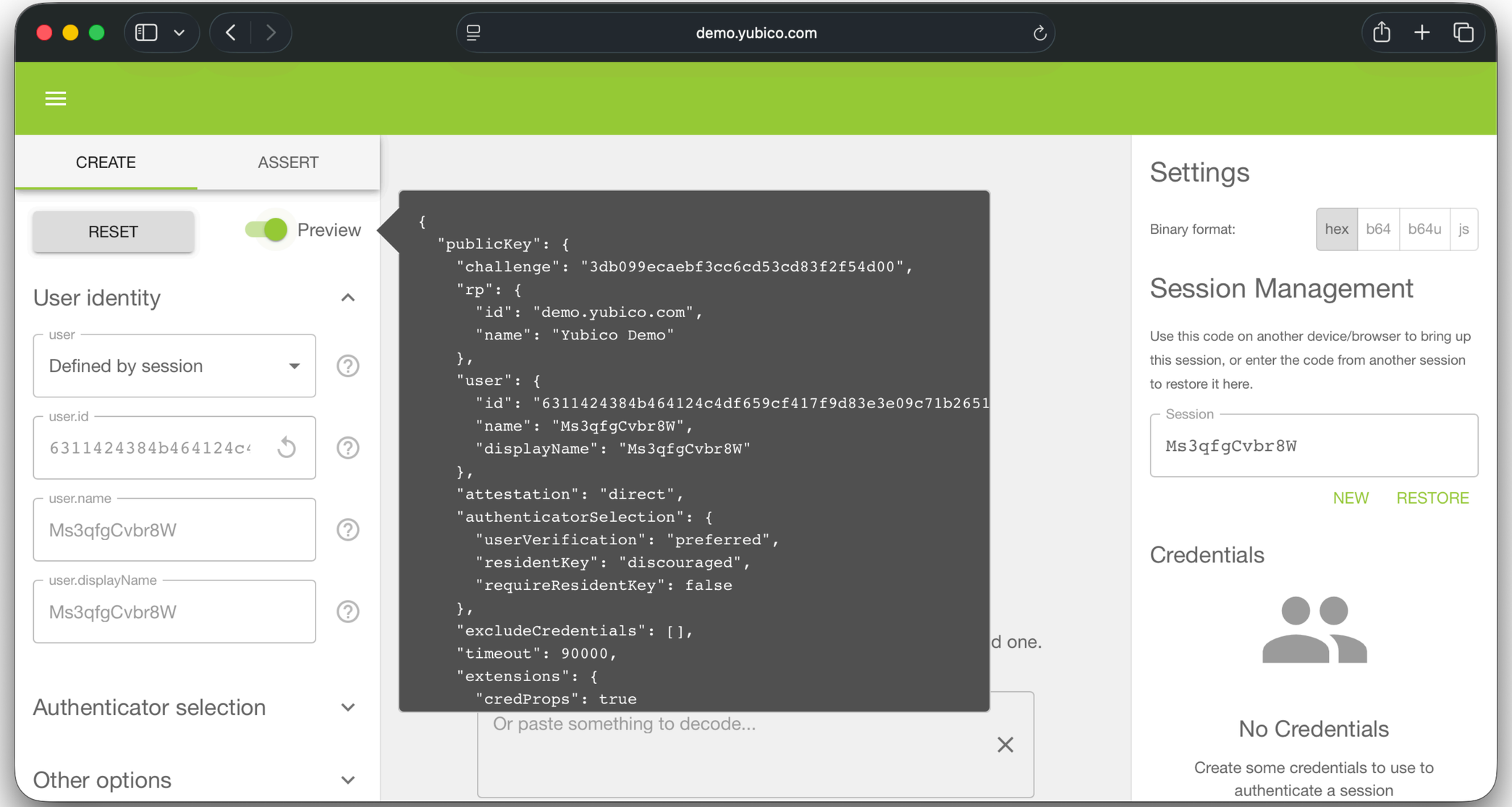
yubico

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



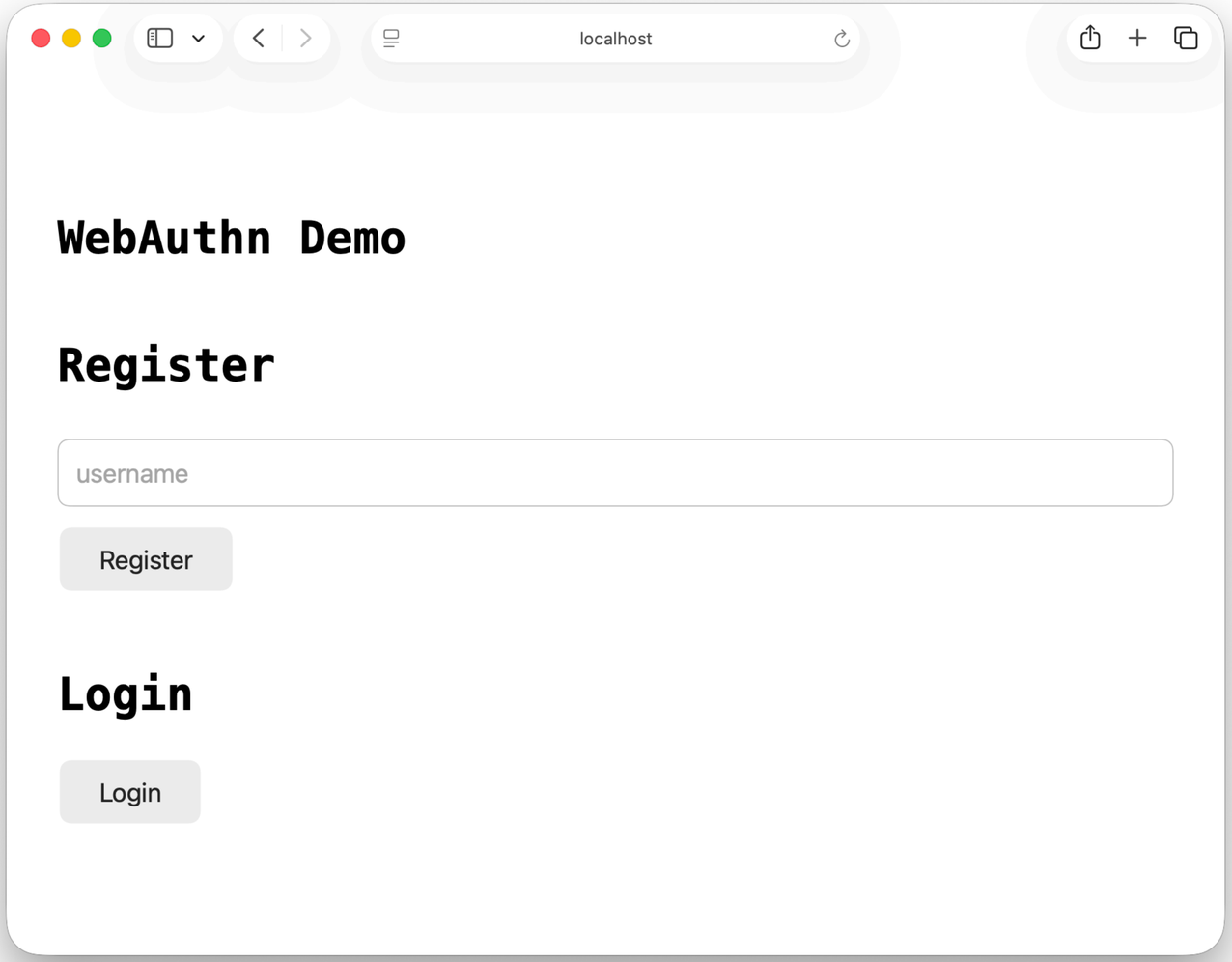
yubico

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



Do It Yourself

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

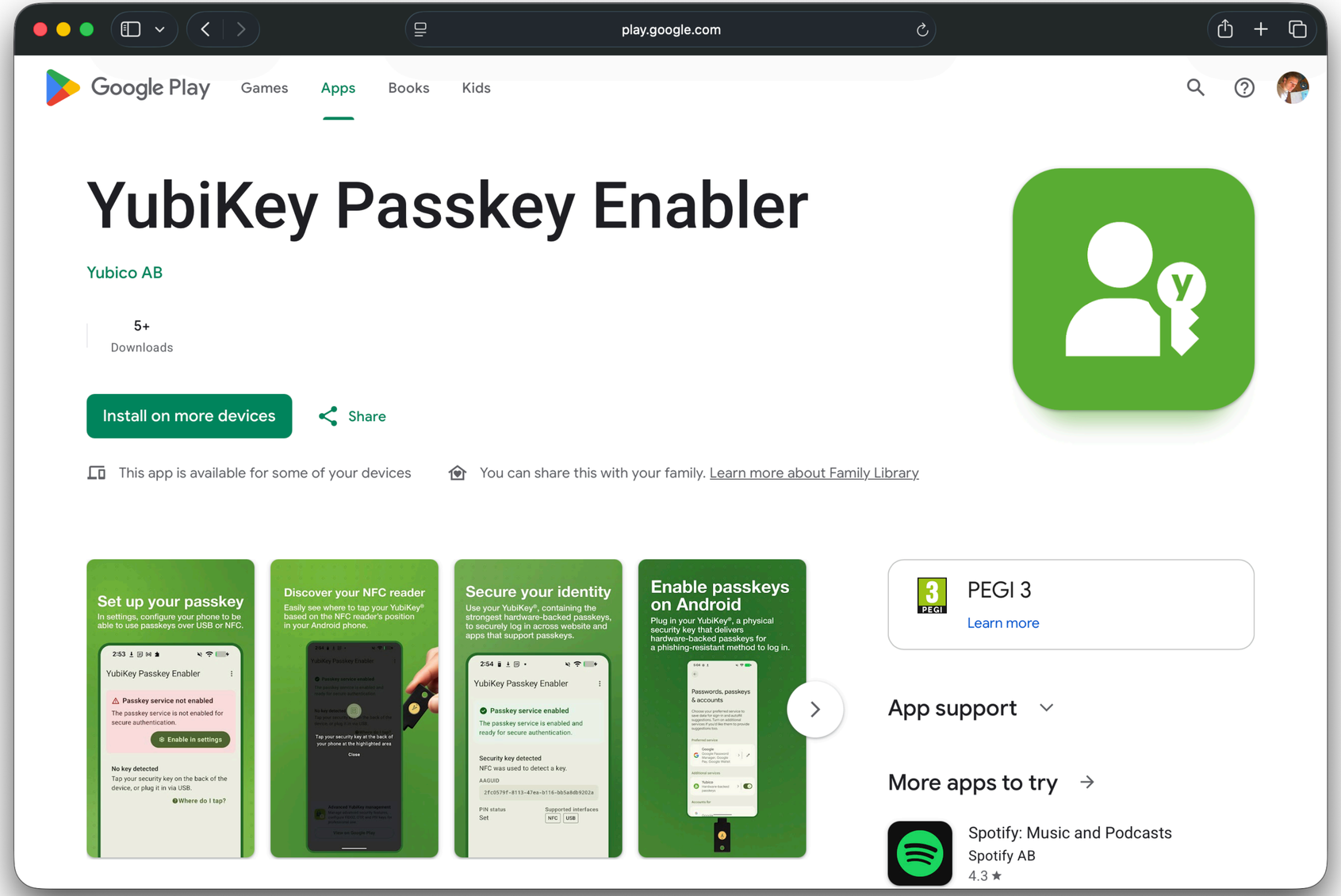


https://tinyurl.com/bsides-passkeys

yubico

Android Passkey Credential Provider Service

- Release Date:
  Mon 22 June 2026



Questions?

WebAuthn: How to get rid of passwords.

yubico