Passkey Workshop

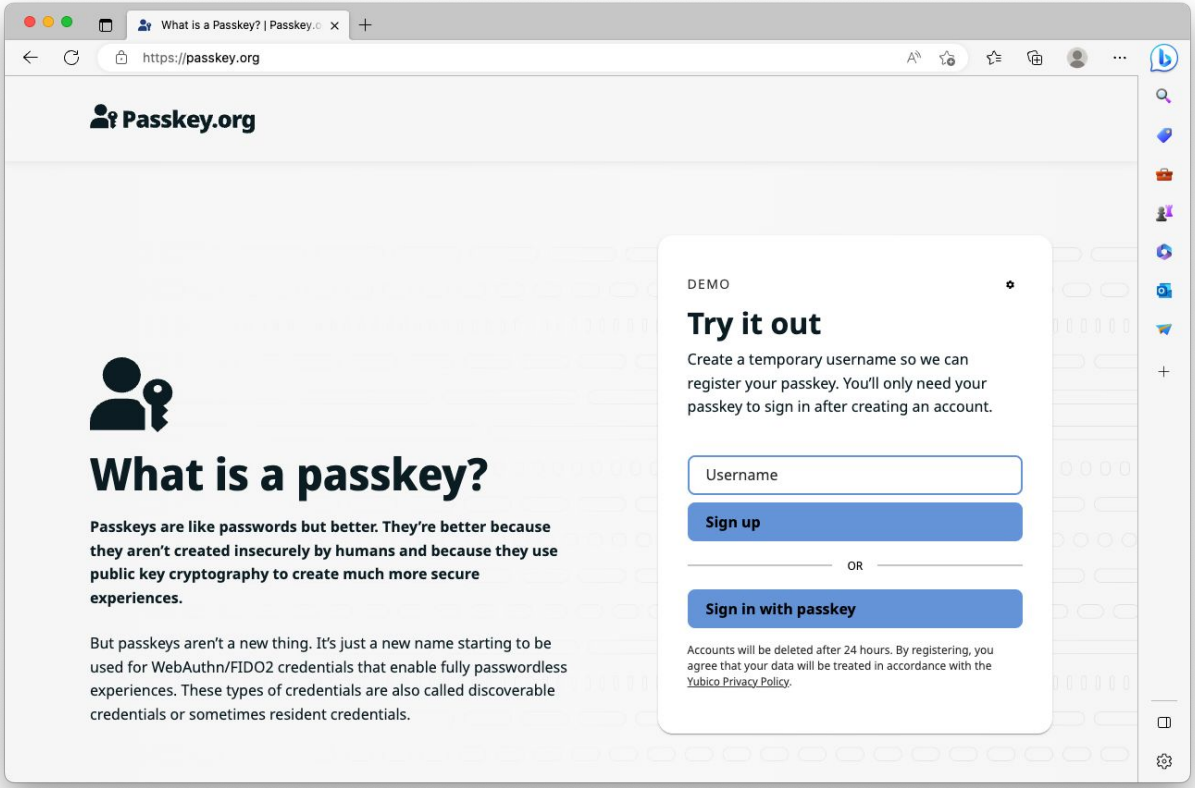
DevFest Berlin - 23rd November 2024

What are passkeys?

- **Passkeys are a more secure alternative to passwords**

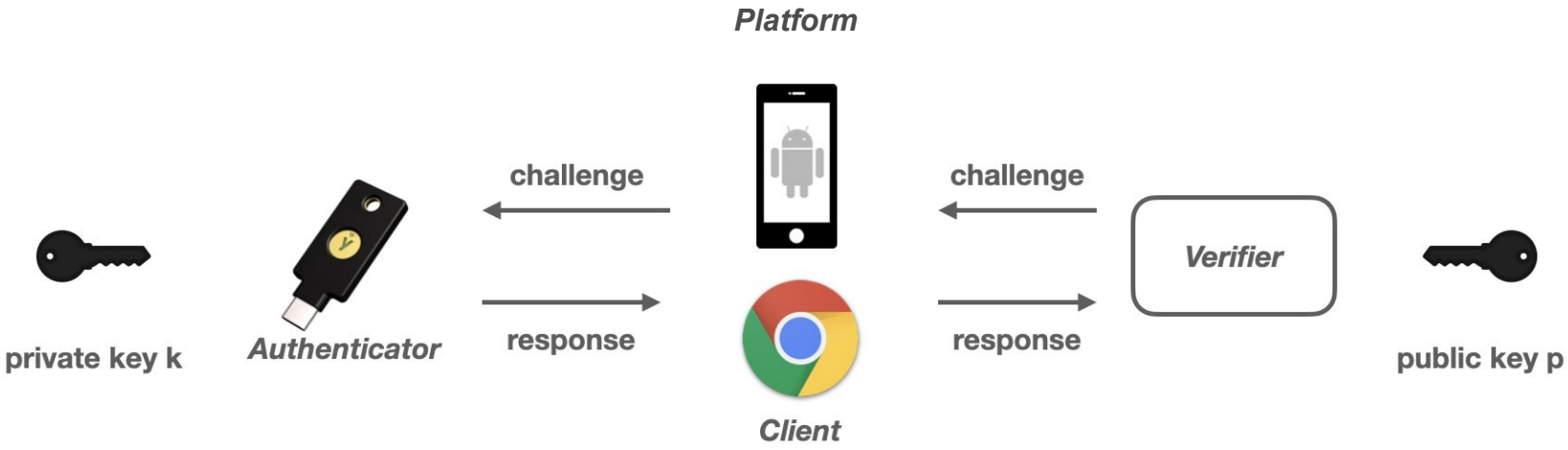
- **More secure, because:**
  - Passkeys are resistant to *phishing*
  - Passkeys have no secrets that can be leaked from servers
  - Passkeys are generated automatically and never reused
  - Passkeys can be protected using secure hardware: security keys
- **Also easier to use:**
  - No need to remember a secret for every account
  - “Sign in with your face, your finger, or your PIN”
  - Optionally, automatically backed up and synced
- **Technically, a passkey is a *FIDO2 credential***

Demo



yubico

Public Key Cryptography



response = sign(k, challenge)

result = verify(p, response, challenge)

FIDO Authenticator

- **Cross-platform Authenticator**
  - aka *roaming authenticator* or FIDO *security key*
  - example: Yubikey
  - Supports *device attestation*
- **Platform Authenticator**
  - Built into devices (phone, laptop, etc)
  - example: Apple TouchID and FaceID
  - example: Windows Hello

- **Authenticators can store multiple passkeys**
- **Cross-platform authenticators can use different transports:**
  - USB, NFC, BLE, Hybrid

Types of Passkeys

Device bound

- Single-device passkeys
- Not copyable; stays on single trusted device (authenticator)
- No device, no access
- Hardware attestation; highly provable security
- Ideal for high assurance use cases
- FIPS Eligible

Syncable | Copyable

- Multi-device passkeys
- Copyable; can be copied to other devices
- Syncable; can be synced to a cloud account
- No hardware attestation
- Ideal for low assurance use cases
- Does not meet FIPS requirements

WebAuthn

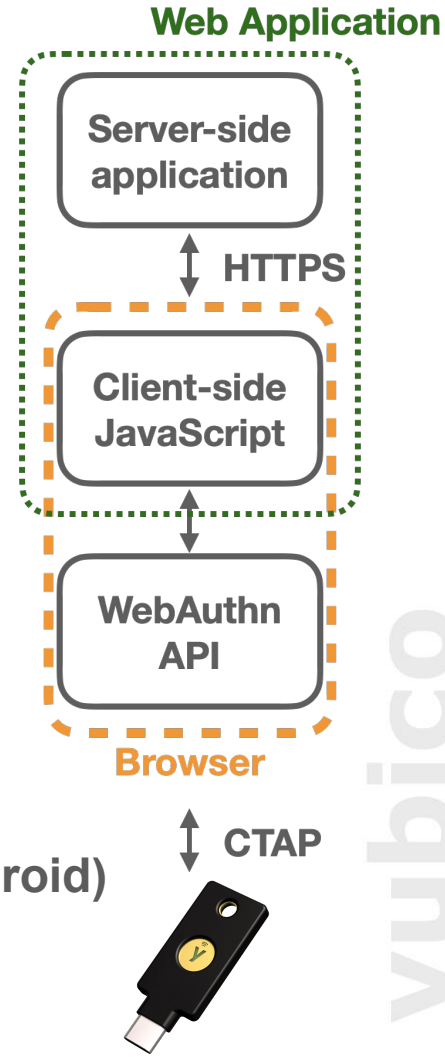
The Web Authentication API (Webauthn)
has two basic methods:

1. `navigator.credentials.create()`
   register a new public key credential
2. `navigator.credentials.get()`
   authenticate using a previously registered credential

See:

https://developer.mozilla.org/en-US/docs/Web/API/Web_Authentication_API

WebAuthn is provided by your Webauthn client
such as your browser (e.g. Chrome) or platform (e.g. Android)



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>
```

Demo

Yubico demo website

demo.yubico.com/webauthn-developers

CREATE | ASSERT

RESET

Preview

User identity

user: Defined by session

user.id: 17af69344cf8639952

user.name: gnPv5FFFrUGK

user.displayName: gnPv5FFFrUGK

Authenticator selection

authenticatorAttachment: *unspecified (default)*

residentKey: discouraged (default)

userVerification: preferred (default)

attestation: direct

WebAuthn Developer Tool

Create a new credential or authenticate with an already registered one.

Or paste something to decode...

Recognized formats:

- PublicKeyCredential: JSON
- attestationObject: binary (CBOR)
- authenticatorData: binary
- WebAuthn client data: JSON binary
- X.509 certificate: PEM binary-wrapped PEM binary (DER)
- JSON: plain binary
- CBOR: binary

Settings

Binary format: hex b64 b64u js

Session Management

Use this code on another device/browser to bring up this session, or enter the code from another session to restore it here.

Session: gnPv5FFFrUGK

NEW RESTORE

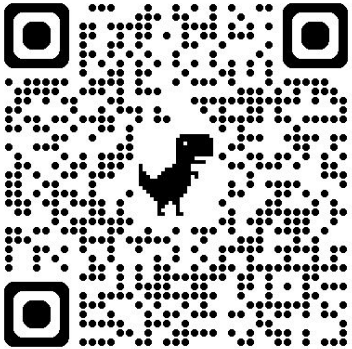
Credentials

No Credentials

Create some credentials to use to authenticate a session

yubico

Do it yourself



- Clone this repository:
  https://github.com/YubicoLabs/webauthn-workshop-starter

- Run a web server so you can access its files on localhost
  For instance:
  ```
  python3 -m http.server 8000
  ```

- Open http://localhost:8000/ in Chrome
  Open Developer Tools to monitor logs

- Register some passkeys
- Experiment with get/create options:
  https://developer.mozilla.org/en-US/docs/Web/API/Web_Authentication_API

Android

may the work begin

yubico

Prerequisites

- Android Studio
- Android device running Android 13+
- YubiKey 5 Series+
- Basic knowledge of Android development
- Basic understanding of WebAuthn concepts
- Clone project:

bit.ly/yubifest24repo

Task 1: Try it out

Test USB YubiKey:

- Build the app
- Connect your YubiKey to the phone
- Look at logs
- Test credential creation
- Test authentication

Test NFC YubiKey:

- Enable NFC on device
- Repeat as above

bit.ly/yubifest24repo

yubico

Task 2: Own Backend

Test USB YubiKey:

- Fork this repository: https://github.com/YubicoLabs/webauthn-workshop-starter
- Publish your repo with GitHub pages (Settings > Pages) https://<you>.github.io/
- Add a Button connecting to that page
- Debug interaction between page and signing

yubico

Troubleshooting

Common issues and solutions:

1. **USB Permission Issues:**
   a. Ensure USB permissions are requested
   b. Check USB configuration
2. **NFC Not Working:**
   a. Verify NFC is enabled
   b. Check NFC configuration
   c. Position YubiKey correctly
3. **PIN Problems:**
   a. Handle PIN retries correctly
   b. Implement proper PIN UI
   c. Handle PIN blocks
   d. Handle no pin set scenario

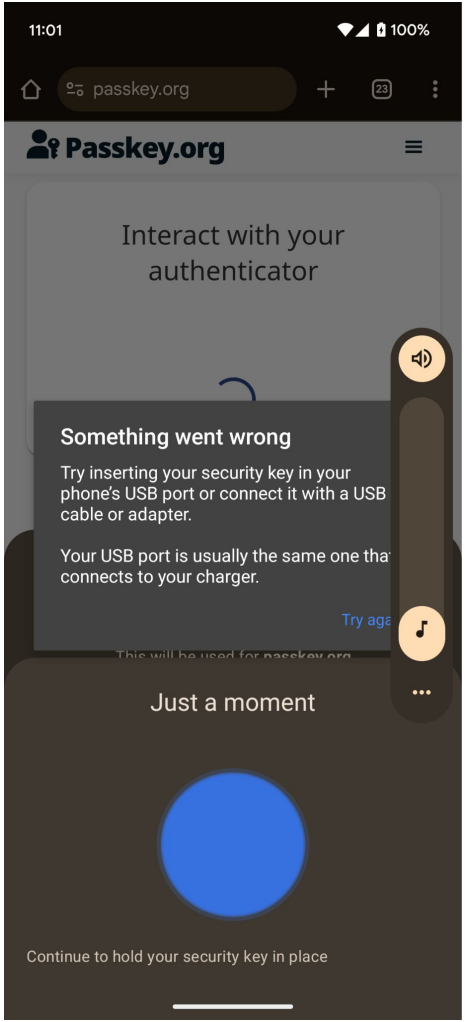
Further Reading

More to Explore

- **Passkey Week @ Google**
- **PGP Signing on Hardware**
  - Github login and commit signing?
  - Emails?
- **Multifactor Authentication**
  - One key to rule all your MFAs
  - (actually two: Have a backup!)
- **typing on touch**
- **Passkey Workshop**
  https://developers.yubico.com/Passkeys/Passkey_workshop.html
- **java-webauthn-server**
  https://developers.yubico.com/java-webauthn-server/

yubico

The key to trust


11:01
100%
passkey.org
Passkey.org
Interact with your authenticator
Something went wrong
Try inserting your security key in your phone's USB port or connect it with a USB cable or adapter.
Your USB port is usually the same one that connects to your charger.
Try again
Just a moment
Continue to hold your security key in place