WebAuthn

Solving Password Fatigue, Phishing, and More

Motivation

Problem: Passwords Suck

- Lots of them to remember
- Phishing
- Memorable =/= good

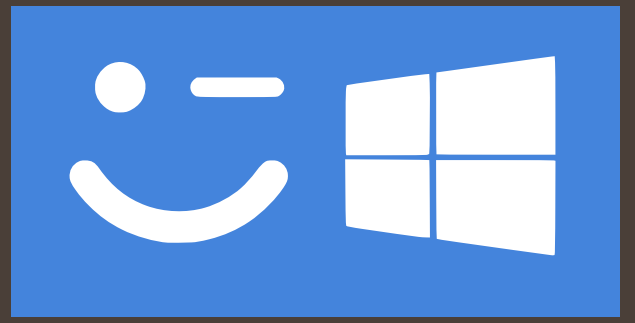
Solution: Don't Use Them

- Slack-style "Magic Link": No passwords, *but*
  - Privacy hazard
  - Email isn't that secure
  - Your email account can't use it
  - (Requires internet access)
  - No standard
- Alternative: WebAuthn

WebAuthn

- Authenticate with public key
- Open standard
- Works offline
- Preserves privacy
- Can use as 1 factor or 2
- Use a physical device to log in

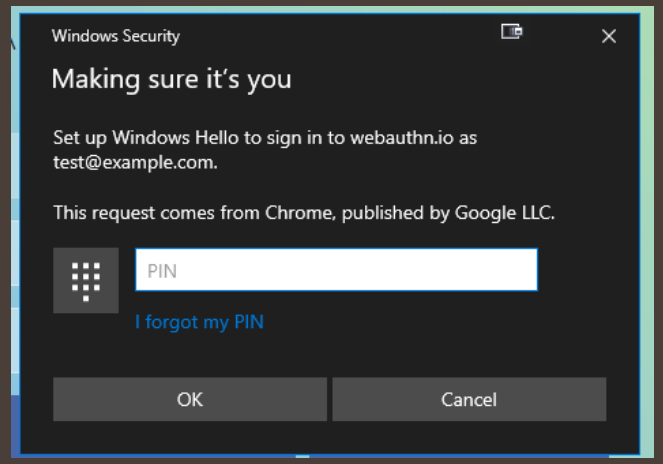
And the best part is that you almost certainly already have a way to use it!



WebAuthn User Experience

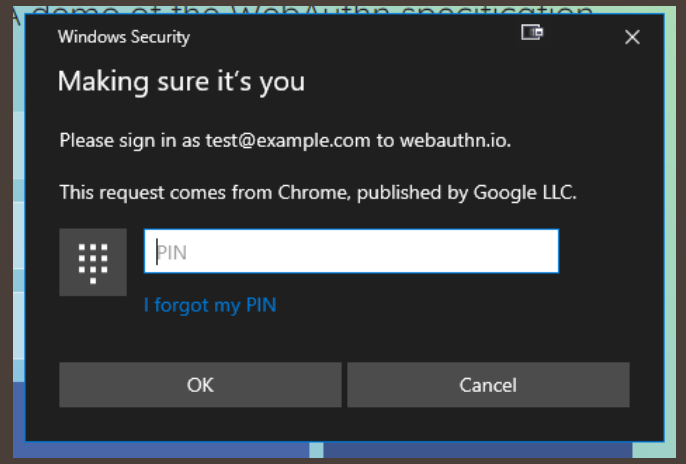
Registration

- Visit Register Page
- Enter email address
- Tap "Register" button
- Authorize security key
- Done



Login

- Visit Login Page
- Enter Email
- Tap "Login" button
- Authorize security key
- Done



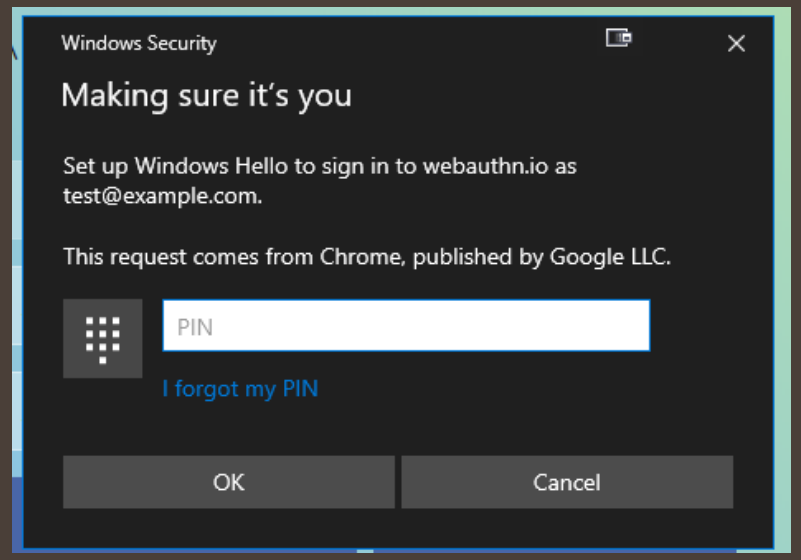
Side Note: WebAuthn PINs

Passwords

- Checked by the receiving server
- Exposed to server (leaks)
- Have to be long to be secure (brute force attacks)

PINs

- Validated by local hardware
- Secure element signs attestation if PIN is correct
- Remote system sees only pubkey
- Secure element has anti-hammering, anti-tampering features



PINs can be short and easy to remember/use without compromising security.

How It Works - Authenticators

- Two kinds of authenticators
  - Cross-platform (Dongle)
  - Platform (TPM/TEE)
- Authenticator has public key
- User must manually authorize all actions
  - Dongle – touch button
  - Platform – software
- Authenticator signatures use ECC in a non-correlatable way
  - That means that a web site can't see that two accounts are owned by the same person
- Authenticator can optionally supply (signed) info about itself
  - High security applications can restrict to known good authenticators

Ways to Authenticate

Authenticator Types

Cross-Platform (Dongle)

- External hardware
- Bluetooth/NFC/USB connection
- Usually just has a button/touch sensor for input
- Can set a PIN

Platform (TPM/TEE)

- Software running on device
- Usually uses platform key storage functionality
- Uses OS authentication (e.g., TouchID/Windows Hello) for authorization

Attestation Types

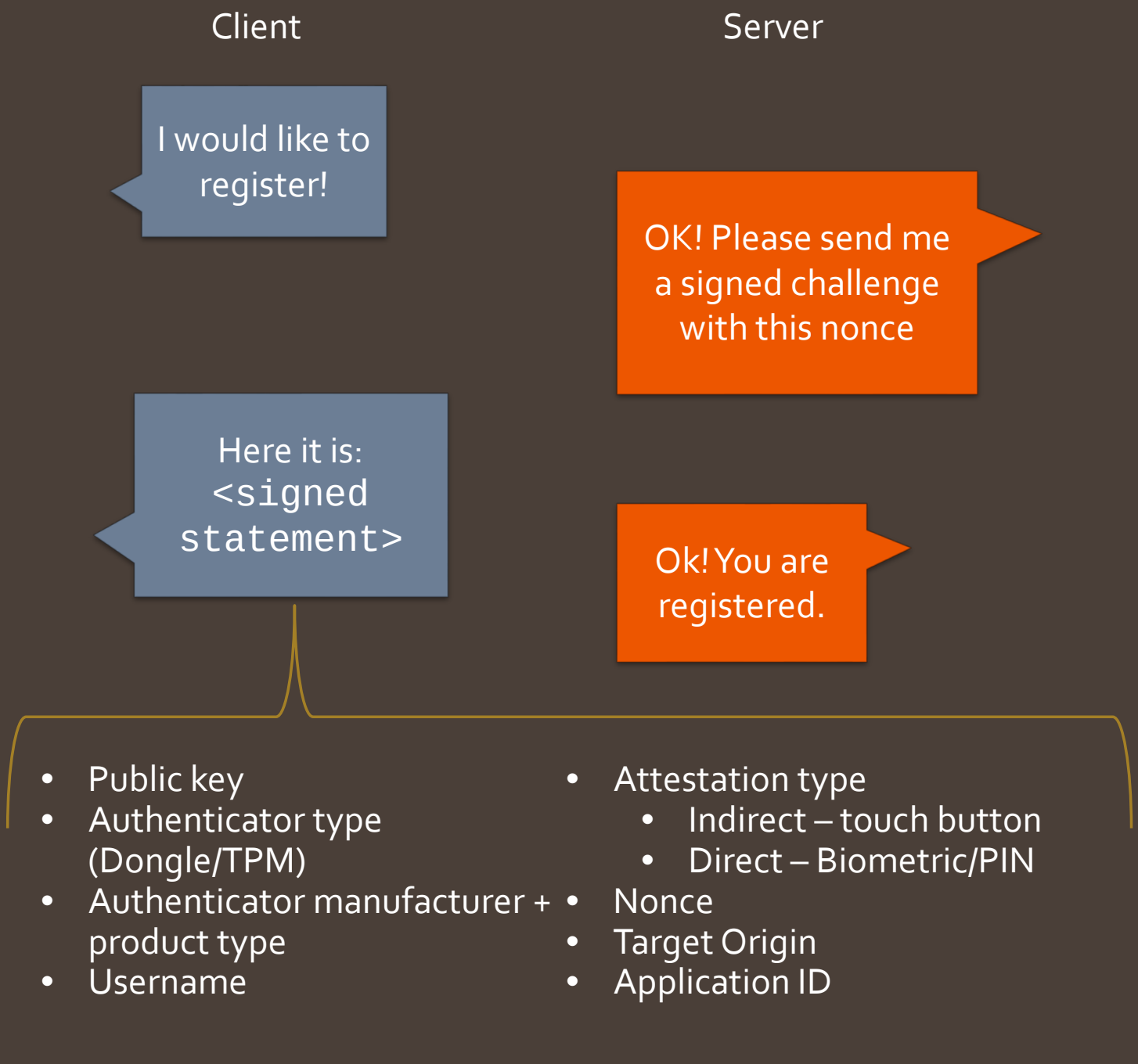
Indirect

- Validates user presence + physical hardware
- Touch sensor on USB dongle
- Button on Bluetooth
- Usually automatic via NFC
- Good only as 1 factor

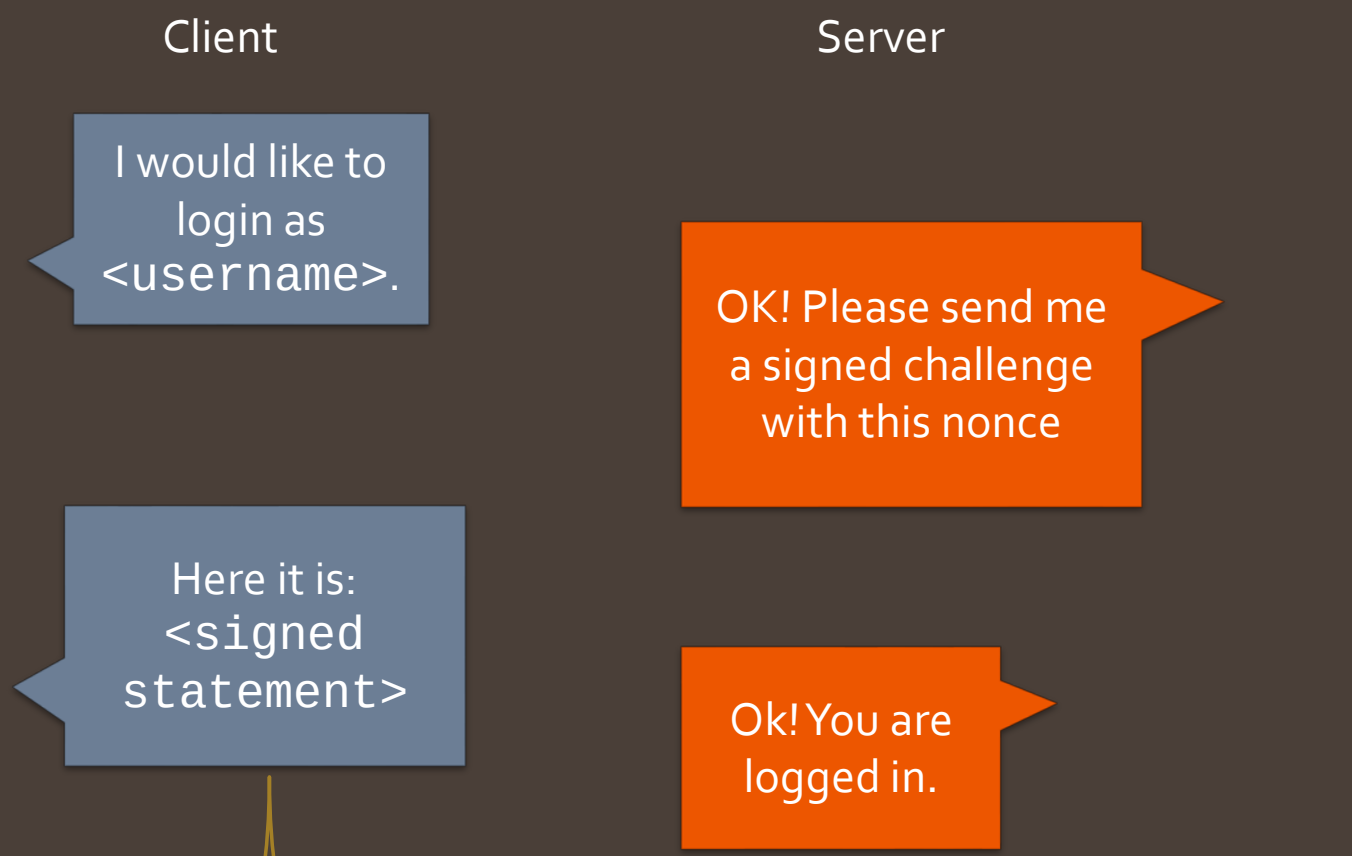
Direct

- Validates user identity
- PIN on (most) external authenticators
- Biometric or PIN for internal ones
- 2 factors inherently – **Allows for passwordless login!**

How It Works – Registration

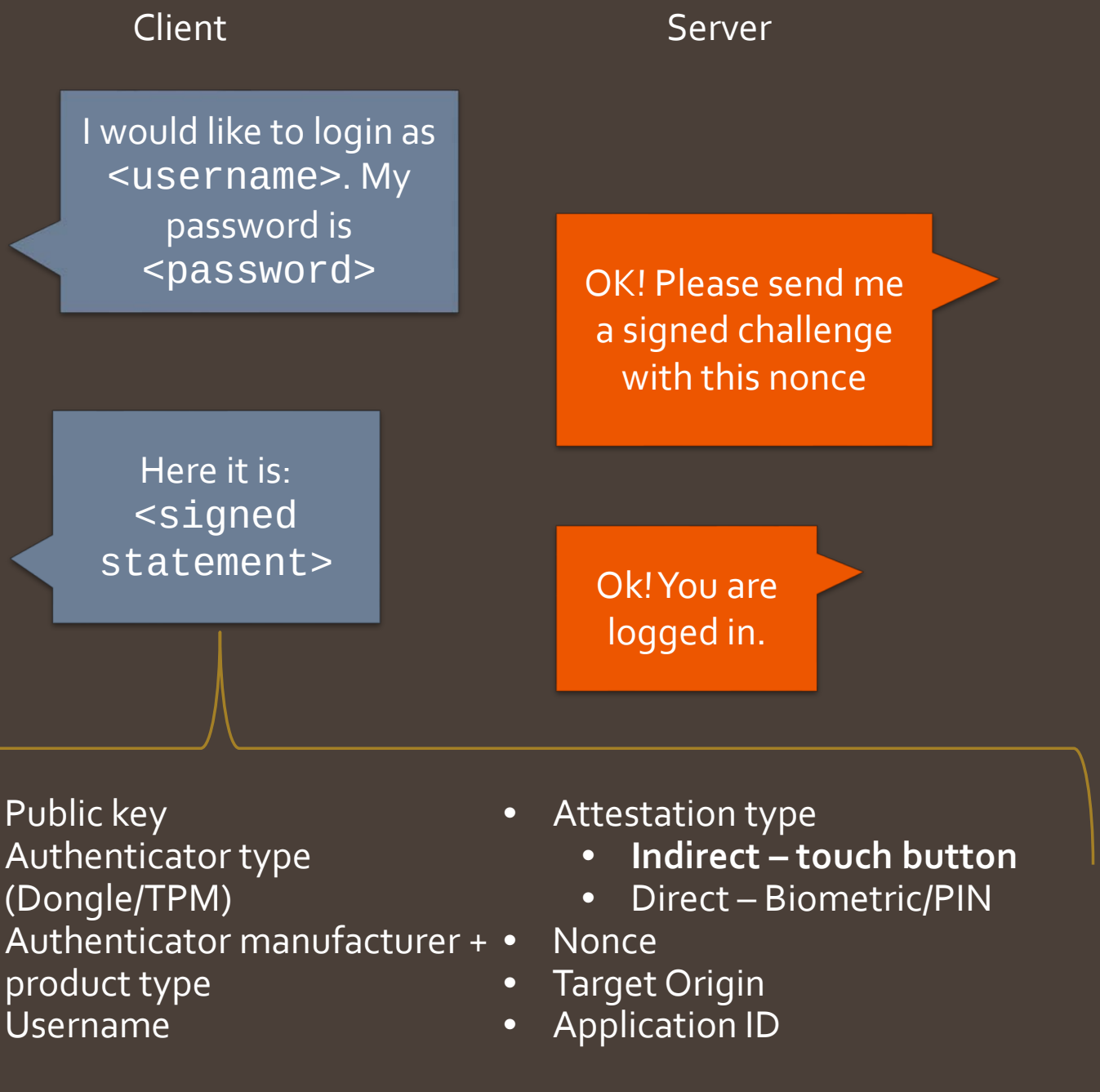


How It Works – Login



- Public key
- Authenticator type (Dongle/TPM)
- Authenticator manufacturer + product type
- Username
- Attestation type
  - Indirect – touch button
  - **Direct – Biometric/PIN**
- Nonce
- Target Origin
- Application ID

How It Works – 2fa



Implementation Considerations

- What if they have multiple devices?
  - Your app must allow multiple authenticators
- What if they lose all of their authenticators?
  - Email recovery?
  - Password backup?
  - Something else?
- Do you do passwordless or 2fa?
- What do you do about users that don't have a dongle/newer device?
- Do you allow all authenticators?
  - Some of the cheaper, lesser known ones almost certainly have flaws
- Is there a support burden? Will users be confused?

Questions?

Thank you.