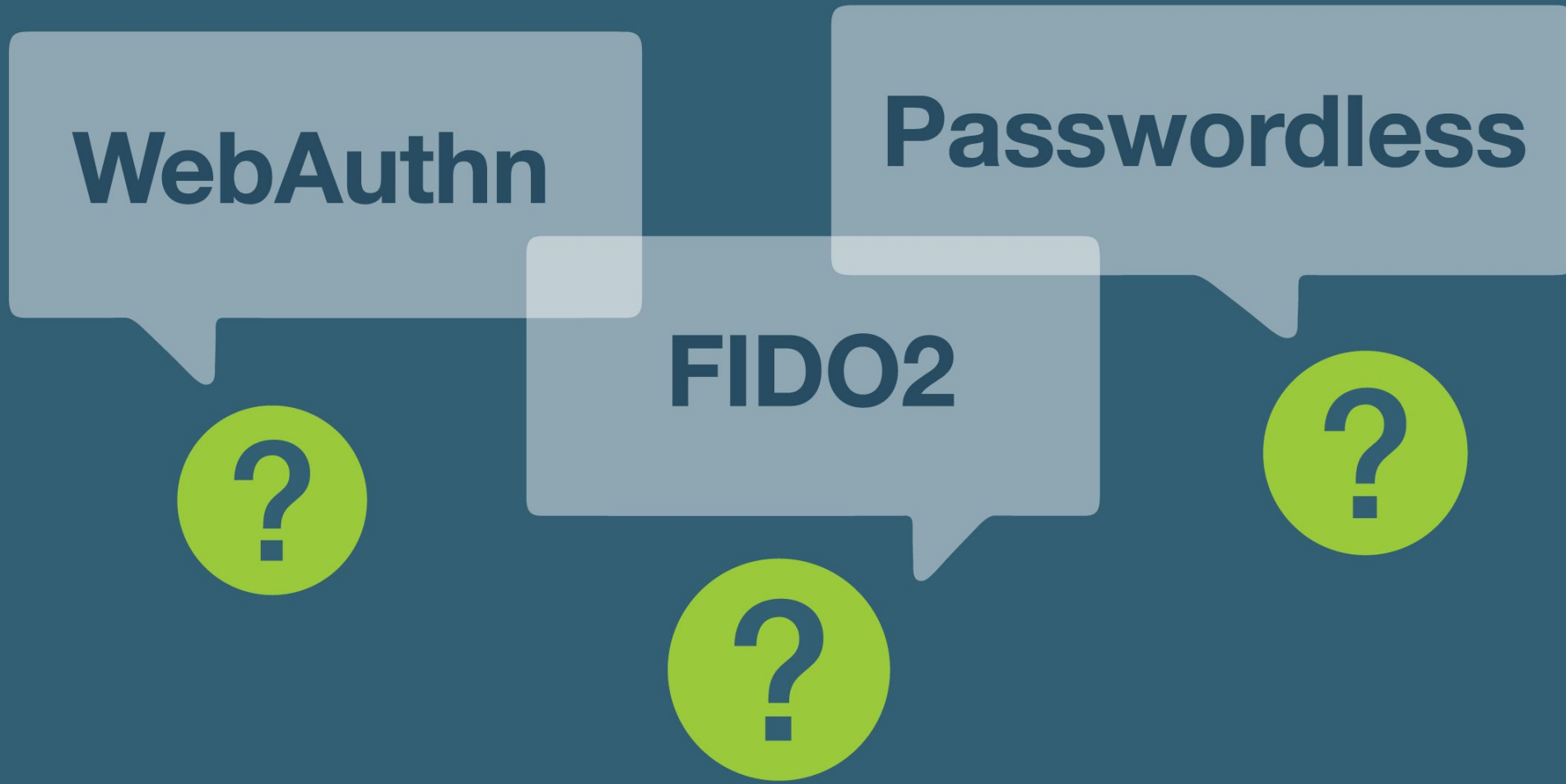
Navigating Passwordless Authentication with FIDO2 & WebAuthn

Jerrod Chong

Chief Solutions Officer, Yubico

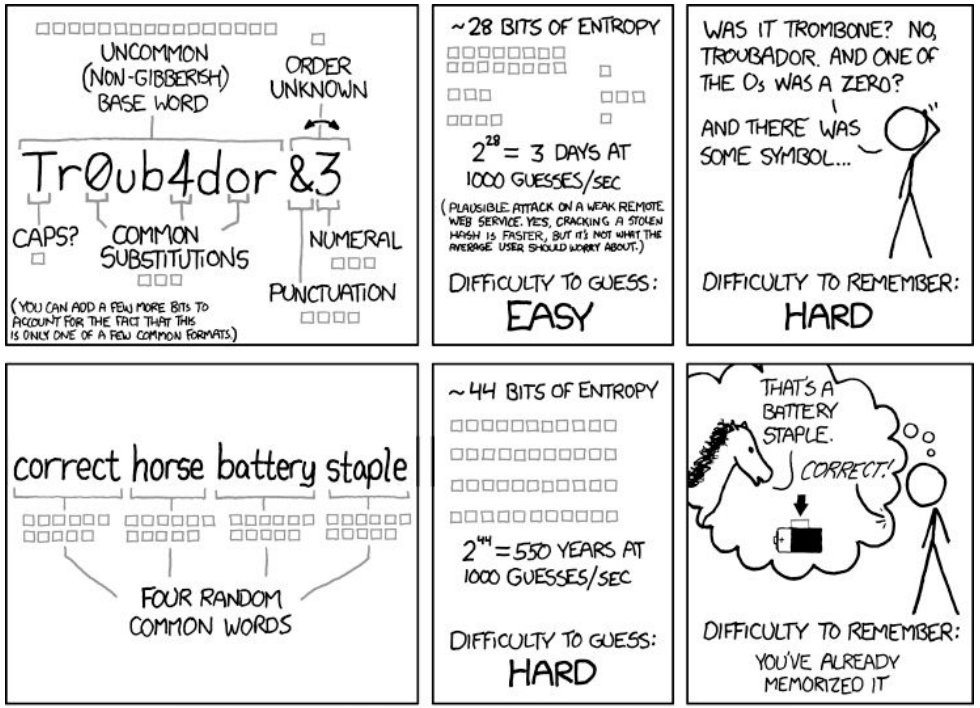
March 3, 2019

yubico


WebAuthn
Passwordless
FIDO2
?
?
?

Passwords

- Hard to remember
- Easy to crack
- Easy to phish
- Many strong passwords take lot's of effort!



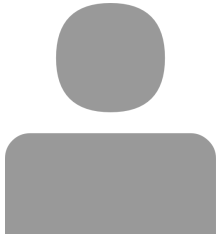
Authentication Factors

Something you **know**

- Password
- PIN

Something you **have**

- Smart card
- OTP dongle
- Mechanical key
- YubiKey



Something you **are**

- Fingerprint
- Face
- Voice
- Iris

Authentication Factors

Pros:

- Can’t be pickpocketed
- Can’t break
- Easy to replace

Cons:

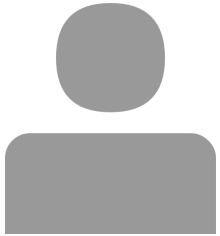
- Easy to steal remotely
- Hard to remember
- Theft is hard to detect

Pros:

- Can’t be stolen remotely
- Theft is easy to detect

Cons:

- Can be pickpocketed
- Can be forgotten, lost or destroyed



Pros:

- Natural to use
- Difficult to lose

Cons:

- Difficult to replace
- May change over time
- Environmental dependencies

Authentication Factors

Pros:

- Can't be pickpocketed
- Can't break
- Easy to replace

Cons:

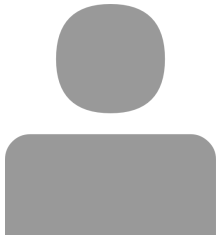
- Easy to steal remotely
- Hard to remember
- Theft is hard to detect

Pros:

- Can't be stolen remotely
- Theft is easy to detect

Cons:

- Can be pickpocketed
- Can be forgotten, lost or destroyed



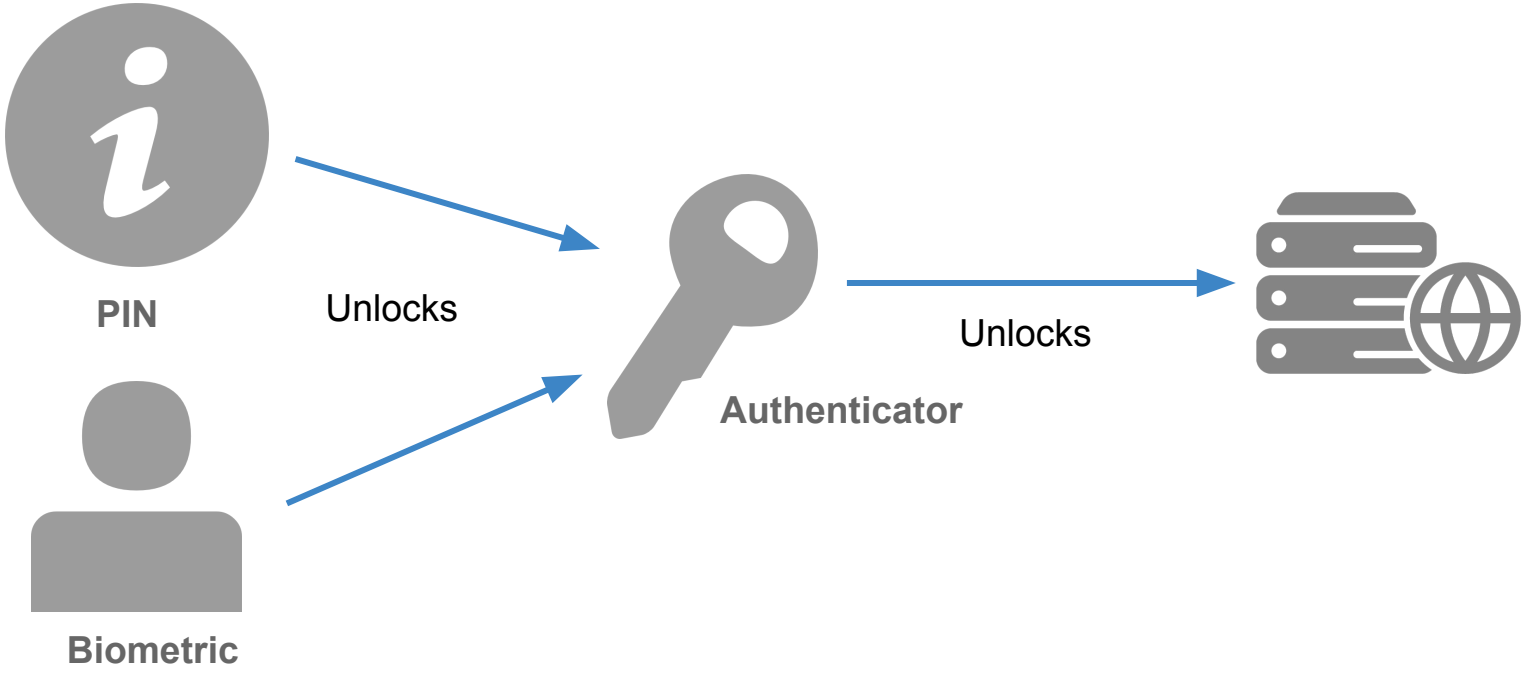
Pros:

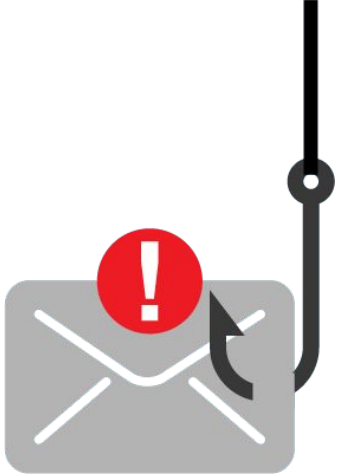
- Natural to use
- Difficult to lose

Cons:

- Difficult to replace
- May change over time
- Environmental dependencies

Multi Factor Authentication

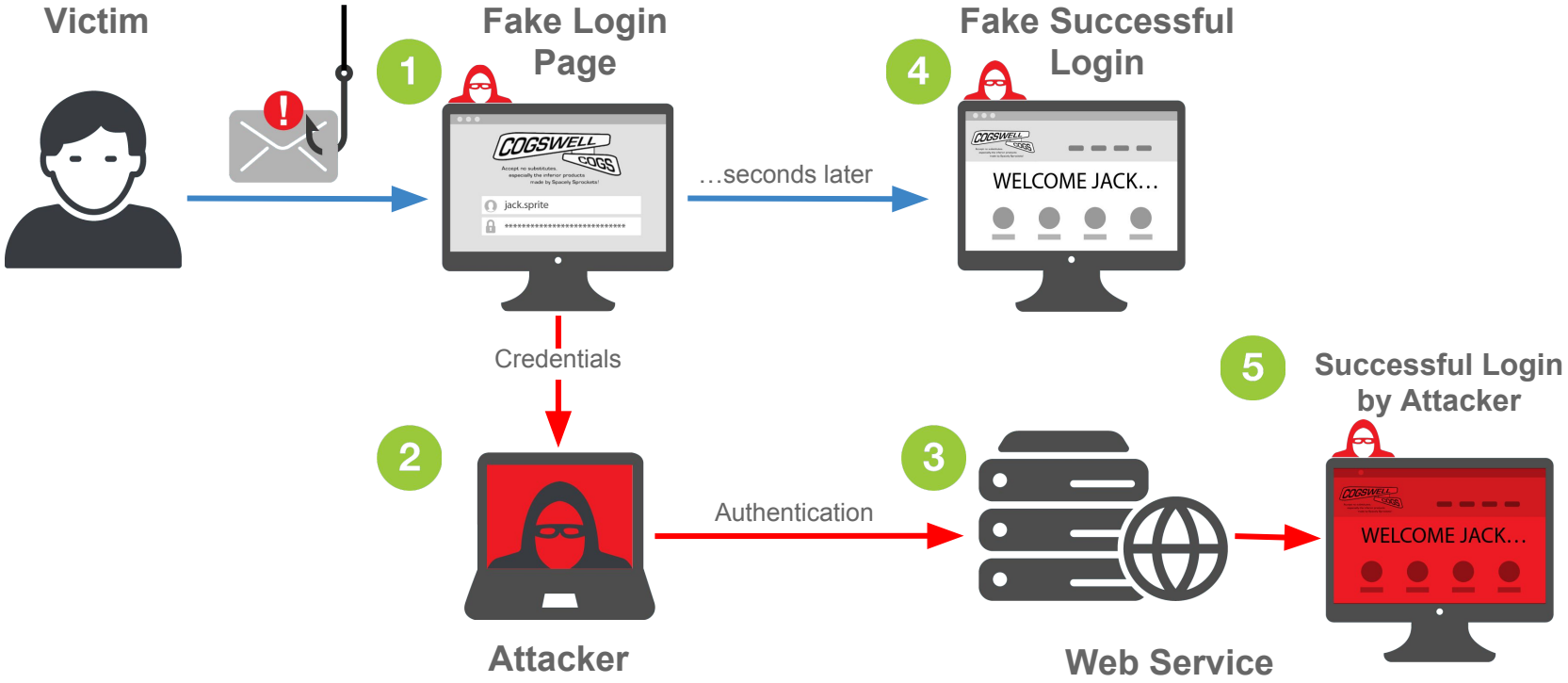




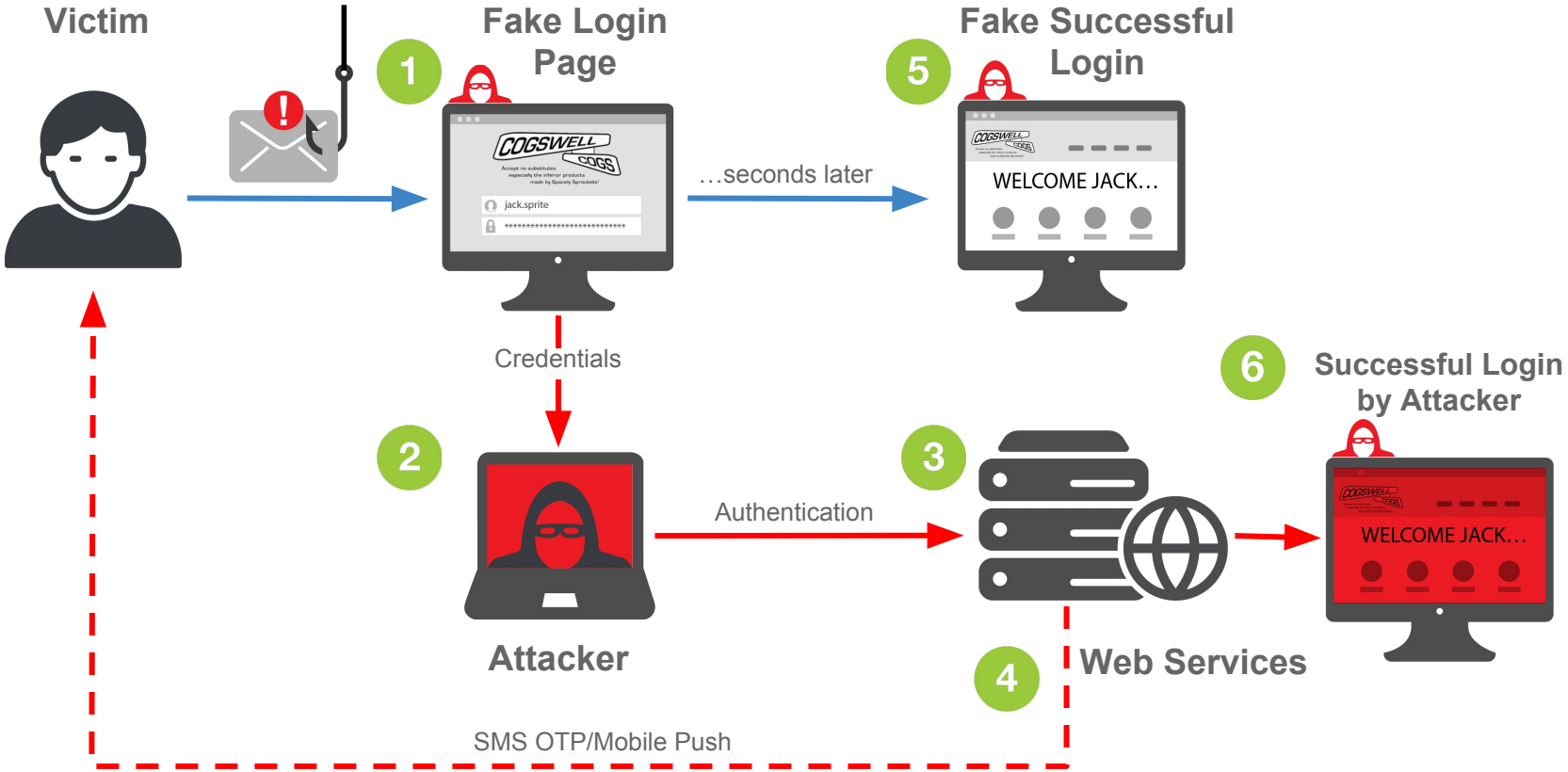
Phishing 101

- Password
- OTP
- Mobile push

How to Phish a Password



How to Phish SMS OTP/Mobile Push



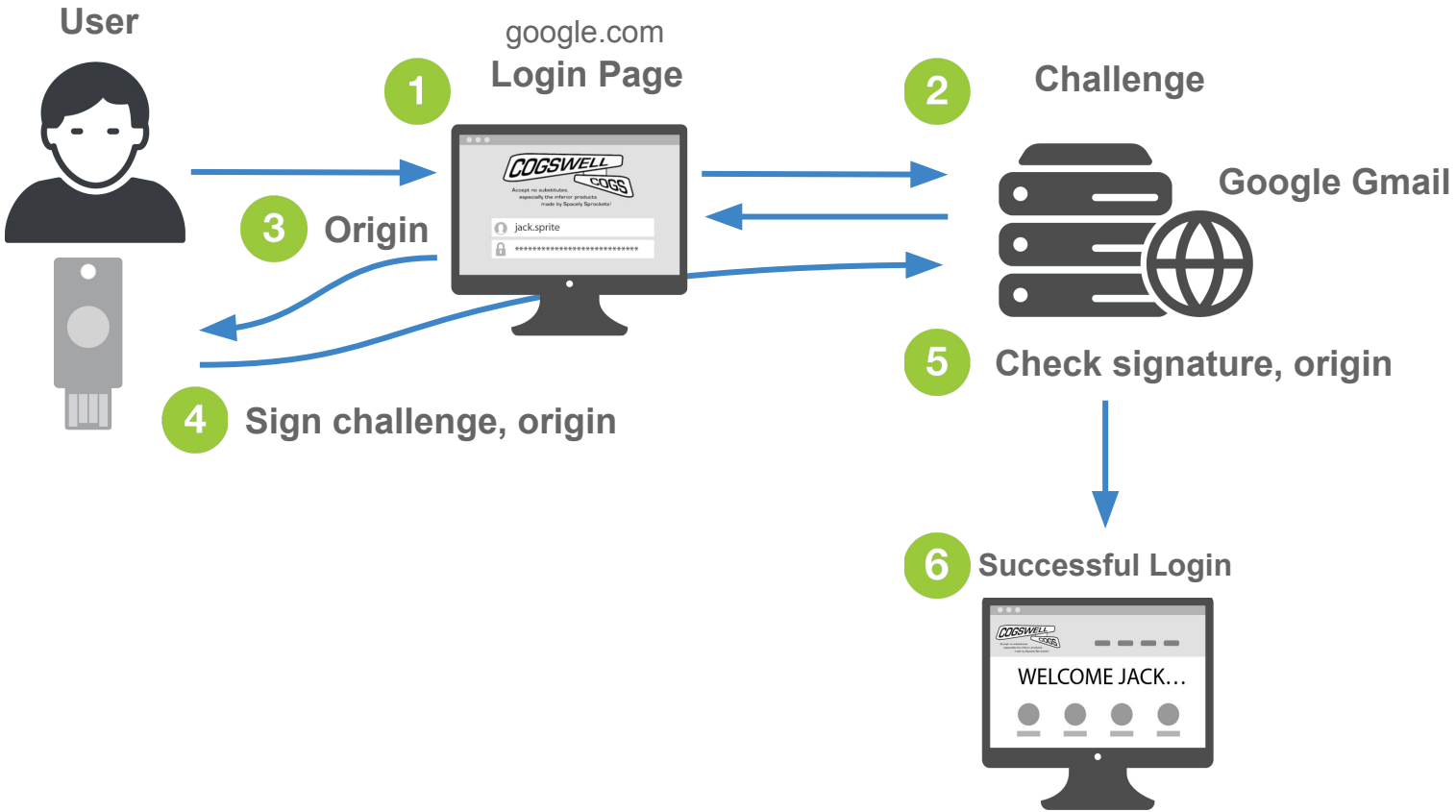
Problem:

Fooling the human is easy

Solution:

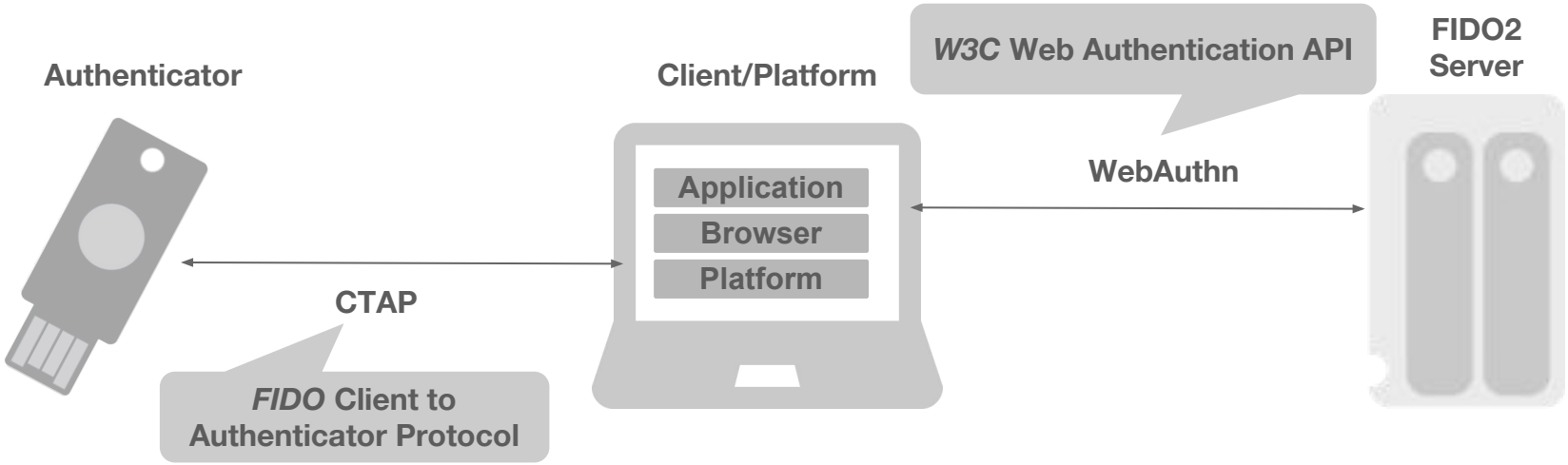
Fooling the computer is hard

How it should work



Check the Origin–Detect Fakes

What is FIDO Authentication?



Open standards utilizing public-key cryptography with phishing protections to enable strong second-factor, first-factor, multi-factor authentication

BSides 2018 - this time last year

WebAuthn specification has attained Candidate Recommendation

WebAuthn supported browsers

- Chrome - intent to ship Chrome 67 (WebAuthn enabled by default)
- Firefox - Firefox 60 (WebAuthn enabled by default)
- Edge - in development

Client To Authenticator Protocol (CTAP) supported platforms

- Android - in development
- Windows 10 - available now

What's New for 2019

WebAuthn to be Ratified as a W3C Standard: March 4, 2019

- W3C will formally promote Web Authentication API "WebAuthn" as a standard
- WebAuthn - standard Web API for services to integrate strong authentication into webapps
- All major browsers supporting/on track to support WebAuthn

FIDO2 Current Status - March 2019

WebAuthn specification has attained Standard

WebAuthn supported browsers

- Chrome - available now!
- Firefox - Firefox 60 (WebAuthn enabled by default)
- Edge - available now!
- Safari - in development

Client To Authenticator Protocol (CTAP) supported platforms

- Android - available now!
- Windows 10 - available now

Why is this a Major Milestone?

Expanded Native Options for User Authentication

Built into the Computer/Phone

Referred to as Platform Authenticators in the WebAuthn specification

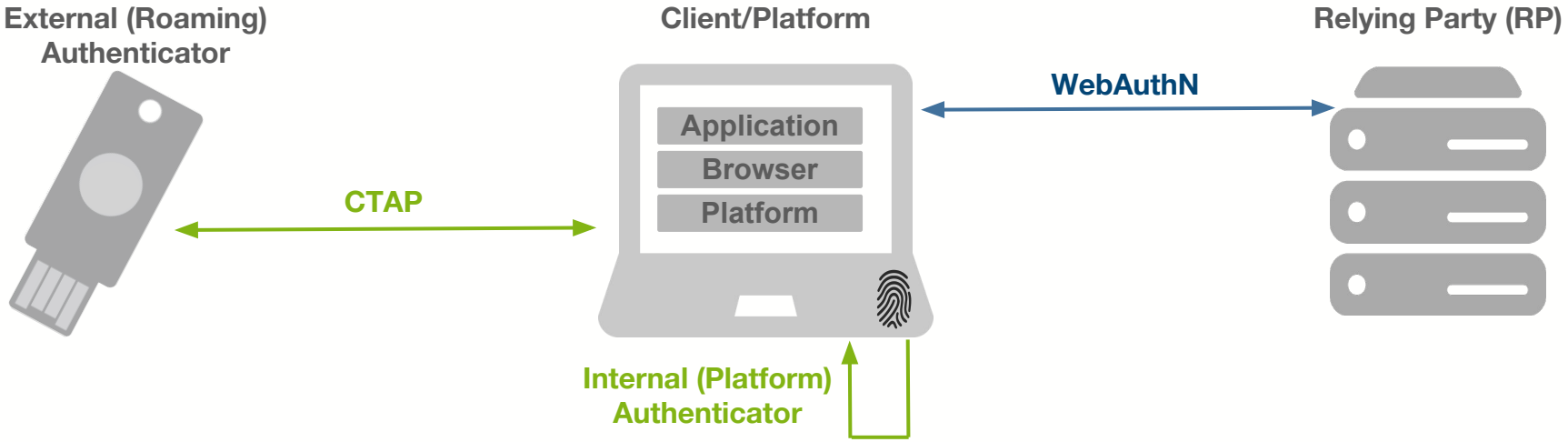
- Biometrics with TPM or TEE/secure enclave
  - Fingerprint Reader
  - Face/Iris/Voice Recognition
- PIN/pattern/passphrase with TPM or TEE/secure enclave

Security Keys

Referred to as Roaming Authenticators in the WebAuthn specification

- Touch sensor with secure element
- PIN and touch sensor with secure element
- Software Authenticators

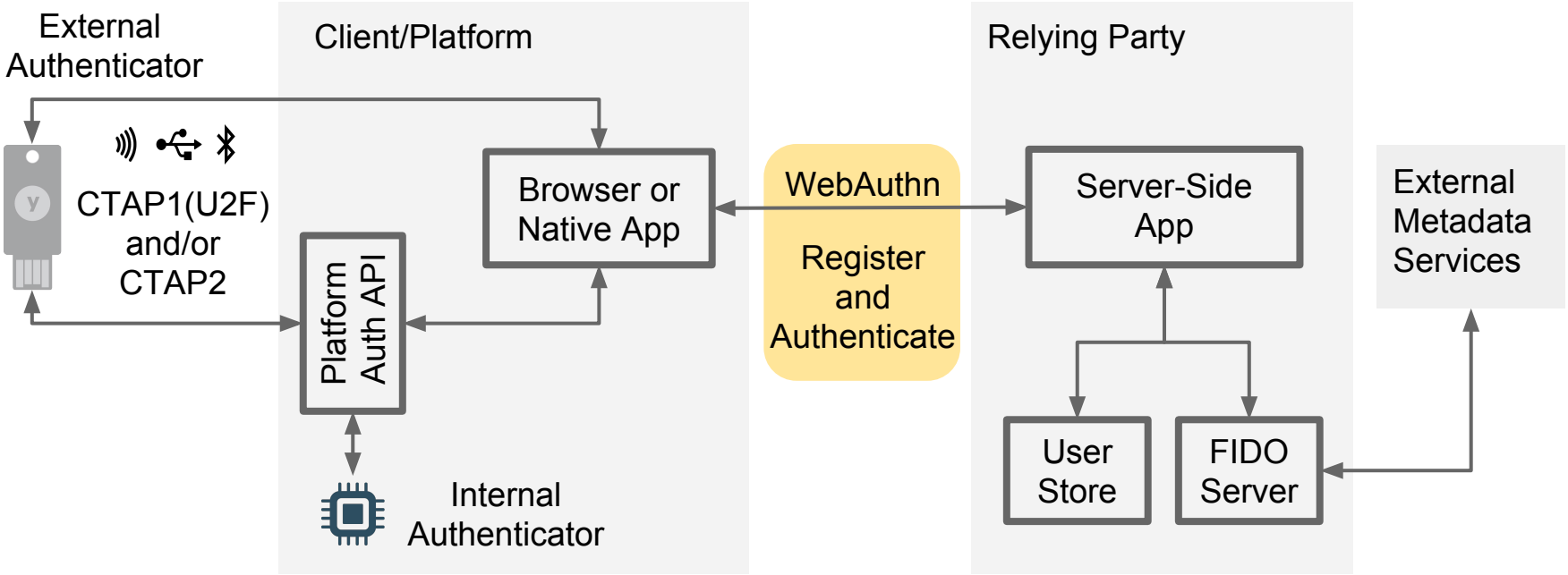
Services Can Offer More Authentication Options



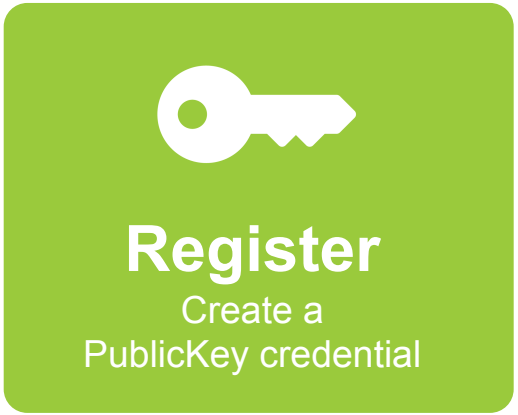
Available now - Android, Chrome on macOS, Edge on Windows, Chrome on Chrome OS

Coming Soon - Firefox on Windows, Firefox on Android, Chrome on Windows

How does it work?

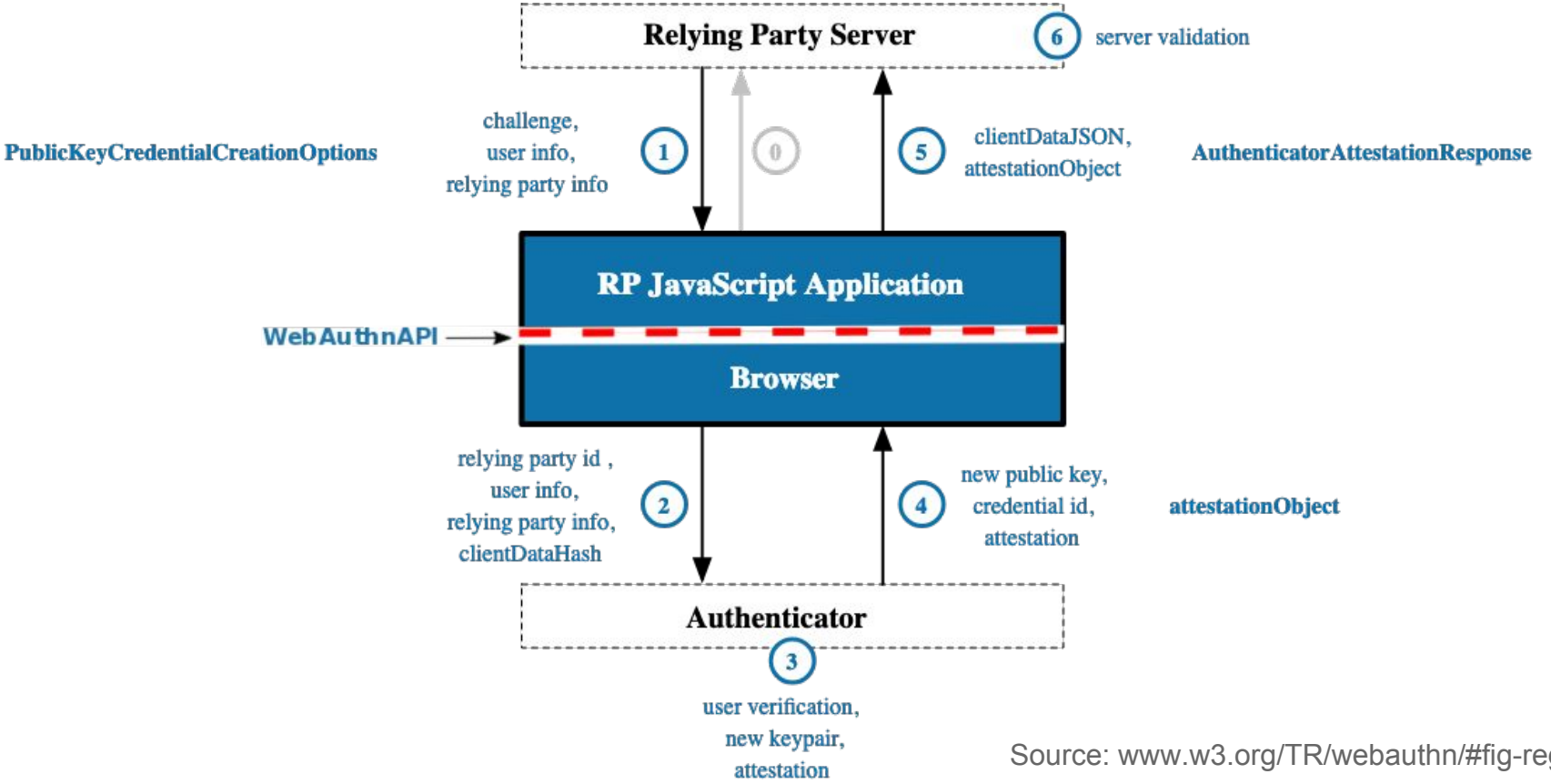


WebAuthn Commands



Allow the website to authenticate the user by asking the user to prove possession of the private key

Registration Flow



Source: www.w3.org/TR/webauthn/#fig-registration

Registration API

- `navigator.credentials.create()` method: https://www.w3.org/TR/webauthn/#createCredential
- Options: https://www.w3.org/TR/webauthn/#dictdef-publickeycredentialcreationoptions

```
dictionary PublicKeyCredentialCreationOptions {
    required PublicKeyCredentialRpEntity         rp;
    required PublicKeyCredentialUserEntity       user;

    required BufferSource                              challenge;
    required sequence<PublicKeyCredentialParameters>   pubKeyCredParams;

    unsigned long                                timeout;
    sequence<PublicKeyCredentialDescriptor>      excludeCredentials = [];
    AuthenticatorSelectionCriteria               authenticatorSelection;
    AttestationConveyancePreference              attestation = "none";
    AuthenticationExtensionsClientInputs         extensions;
};
```

Registration API

- `navigator.credentials.create()` method: https://www.w3.org/TR/webauthn/#createCredential
- Options: https://www.w3.org/TR/webauthn/#dictdef-publickeycredentialcreationoptions

```
dictionary PublicKeyCredentialCreationOptions {
    required PublicKeyCredentialRpEntity         rp;
    required PublicKeyCredentialUserEntity       user;

    required BufferSource                              challenge;
    required sequence<PublicKeyCredentialParameters>   pubKeyCredParams;

    unsigned long                                timeout;
    sequence<PublicKeyCredentialDescriptor>      excludeCredentials = [];
    AuthenticatorSelectionCriteria               authenticatorSelection;
    AttestationConveyancePreference              attestation = "none";
    AuthenticationExtensionsClientInputs         extensions;
};
```

Creating Cross-Platform Credential

- **Requires option `authenticatorAttachment: cross-platform` at registration**
- **Combine with User Presence**
  - Set option `userVerification: “discouraged”` **to use as 2nd factor**

```
dictionary AuthenticatorSelectionCriteria {
    AuthenticatorAttachment      authenticatorAttachment;
    boolean                      requireResidentKey = false;
    UserVerificationRequirement  userVerification = "discouraged";
};
```

```
enum AuthenticatorAttachment {
    "platform",
    "cross-platform"
};
```

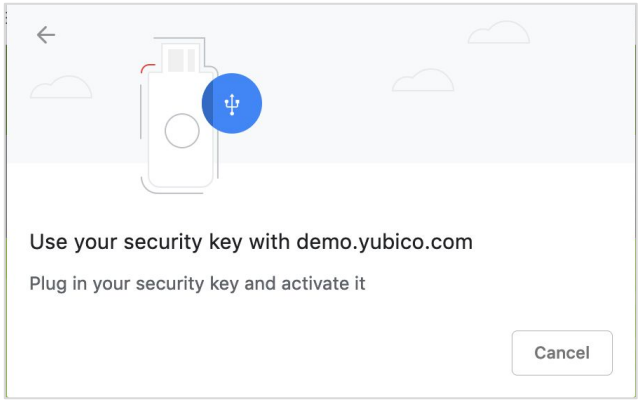
Cross-Platform Credential Creation Example

PublicKeyCredentialCreationOptions

```
{
  "publicKey": {
    "attestation": "direct",
    "authenticatorSelection": {
      “authenticatorAttachment”: “cross-platform”,
      "requireResidentKey": false,
      "userVerification": "discouraged"
    },
    "challenge": "qNqrdXUrk5S7dCM1MAY...",
    "excludeCredentials": [],
    "pubKeyCredParams": [{
        "alg": -7,
        "type": "public-key"
      }],
    "rp": {
      "id": "demo.yubico.com",
      "name": "YubicoDemo"
    },
    "user": {
      "displayName": "Yubico demo user",
      "id":
"bz9ZDfHzOBLycqISTAdWwWIZt8VO-6mT3hBNXS5jwmY="
    }
  }
}
```

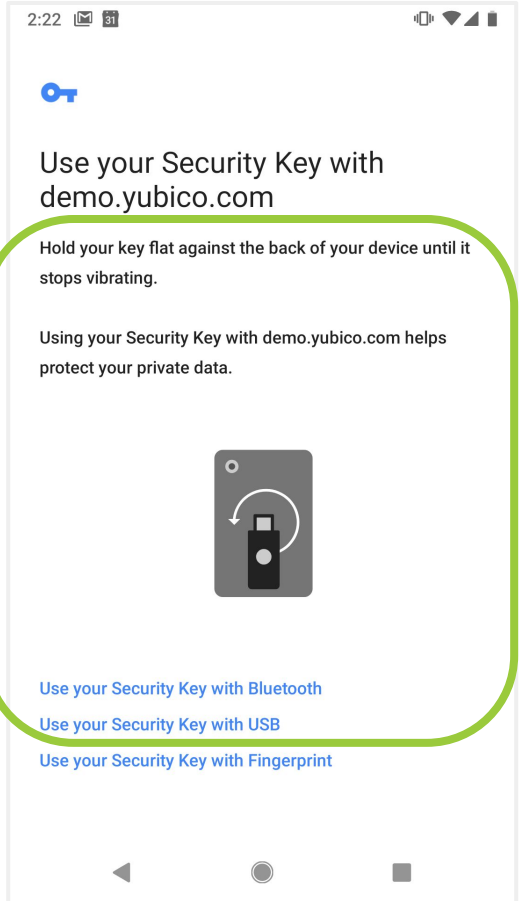
Windows - USB

macOS - USB



Cross-Platform Credential Creation Example

Android - NFC, BLE, USB



Availability of Platform Authenticator

- RP who wants users to register specifically with User-Verifying Platform Authenticator can request
  - `PublicKeyCredential.isUserVerifyingPlatformAuthenticatorAvailable()`
- Client returns true or false

```
partial interface PublicKeyCredential {
    static Promise<boolean> isUserVerifyingPlatformAuthenticatorAvailable();
};
```

Creating Platform Credential

- **Requires option `authenticatorAttachment: platform` at registration**
- **Combine with User Verification for built-in MFA**
  - Set option `userVerification: “preferred” or “required”`

```
dictionary AuthenticatorSelectionCriteria {
    AuthenticatorAttachment      authenticatorAttachment;
    boolean                      requireResidentKey = false;
    UserVerificationRequirement  userVerification = "preferred";
};
```

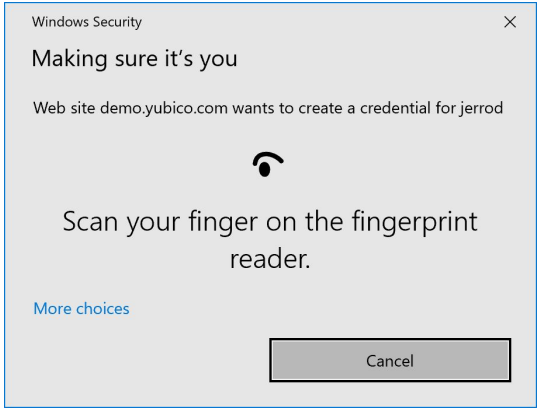
```
enum AuthenticatorAttachment {
    "platform",
    "cross-platform"
};
```

Platform Credential Creation Example

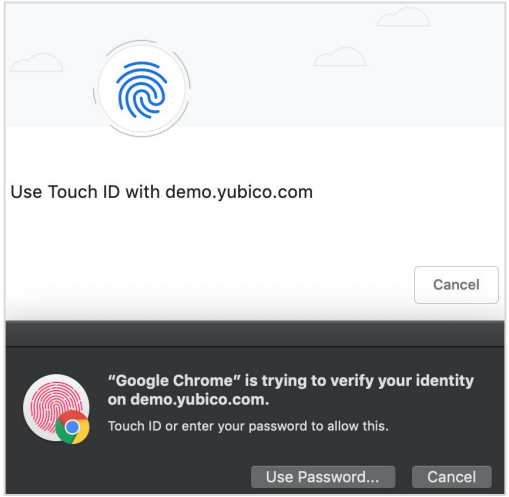
PublicKeyCredentialCreationOptions

```
{
  "publicKey": {
    "attestation": "direct",
    "authenticatorSelection": {
      “authenticatorAttachment”: “platform”,
      "requireResidentKey": false,
      "userVerification": "preferred"
    },
    "challenge": "qNqrdXUrk5S7dCM1MAY...",
    "excludeCredentials": [],
    "pubKeyCredParams": [{
        "alg": -7,
        "type": "public-key"
      }],
    "rp": {
      "id": "demo.yubico.com",
      "name": "YubicoDemo"
    },
    "user": {
      "displayName": "Yubico demo user",
      "id": "bz9ZDfHzOBLycqISTAdWwWIZt8VO-6mT3hBNXS5jwmY="
    }
  }
}
```

Windows - Hello Fingerprint

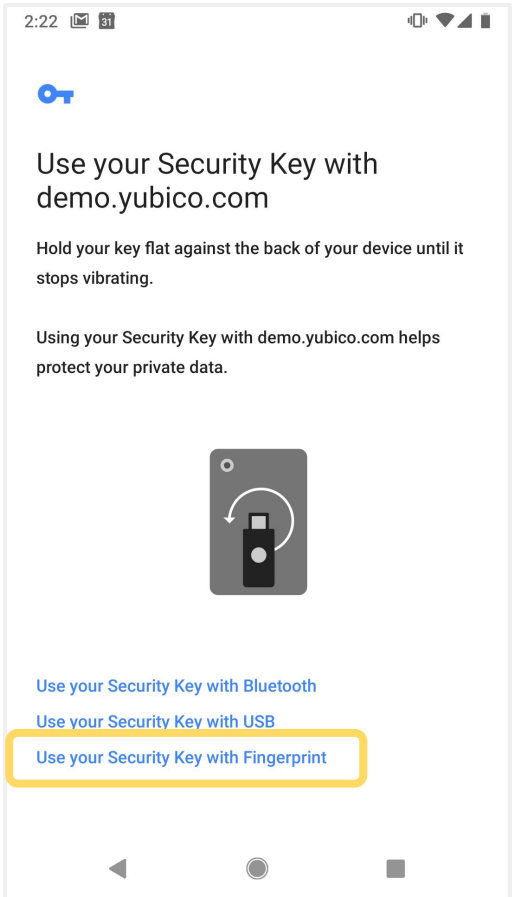


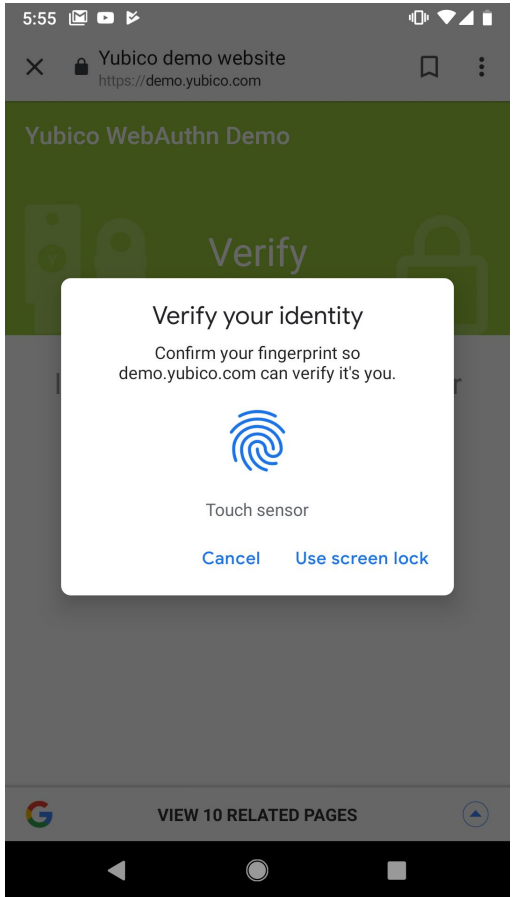
macOS - Touch ID

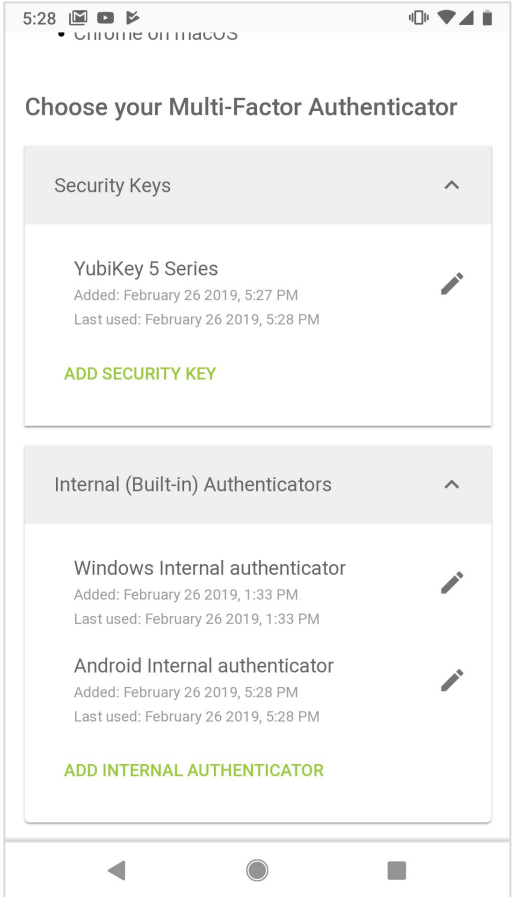


Platform Credential Creation Example

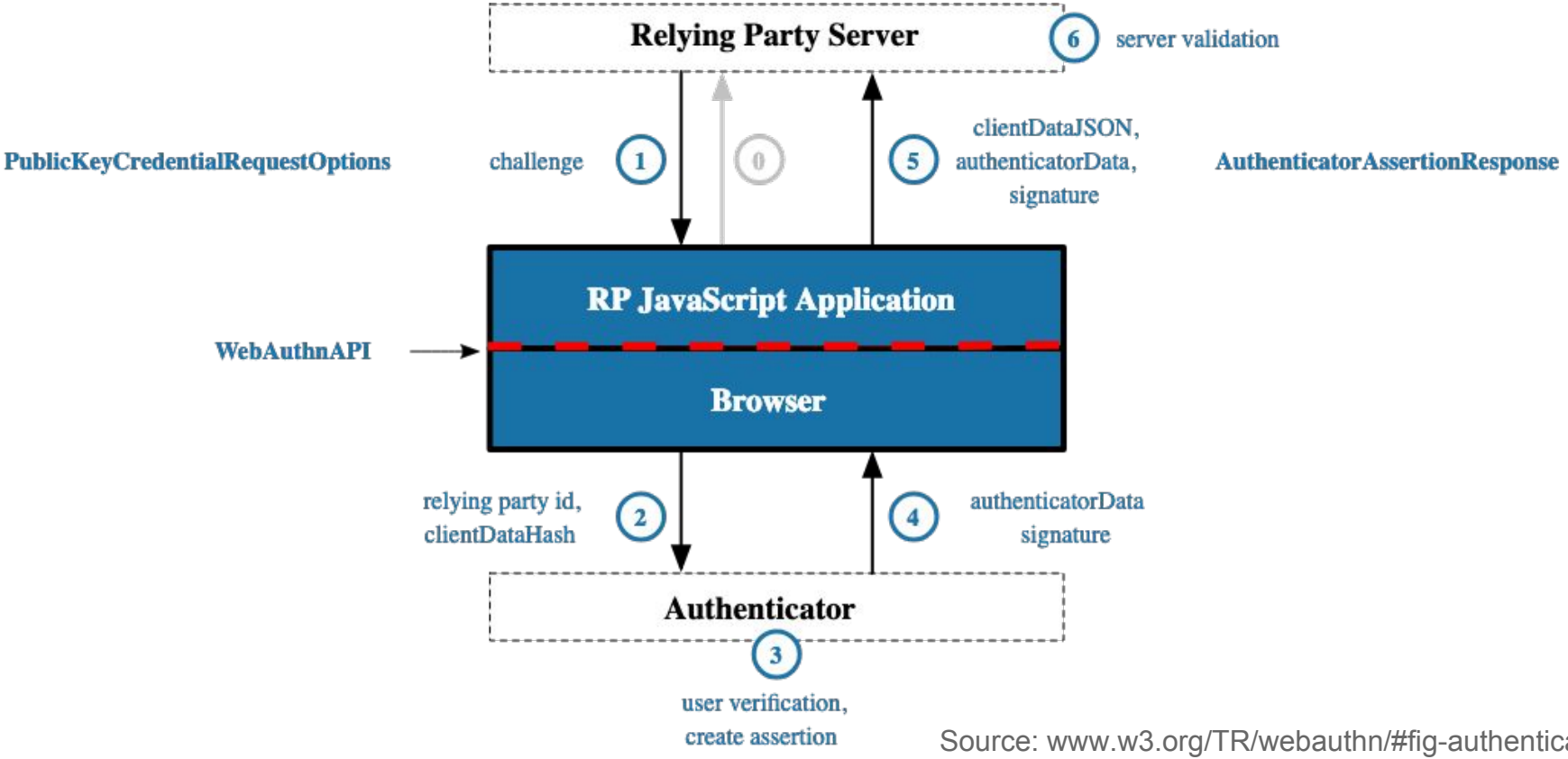
Android - Fingerprint







Authentication Flow



Source: www.w3.org/TR/webauthn/#fig-authentication

Authentication API

- `navigator.credentials.get()` method: https://www.w3.org/TR/webauthn/#getAssertion
- Options: https://www.w3.org/TR/webauthn/#dictdef-publickeycredentialrequestoptions

```
dictionary PublicKeyCredentialRequestOptions {
    required BufferSource                challenge;
    unsigned long                        timeout;
    USVString                            rpId;
    sequence<PublicKeyCredentialDescriptor> allowCredentials = [];
    UserVerificationRequirement          userVerification = "preferred";
    AuthenticationExtensionsClientInputs extensions;
};
```

Authenticator Transports

- **Relying Party can hint how to communicate with the authenticator**
  - `allowCredentials[i].transports`
- **Four options are available** `transports: ["usb", "nfc", “ble”, “internal”]`
- **`transports` informs client UI to optimize user experience**

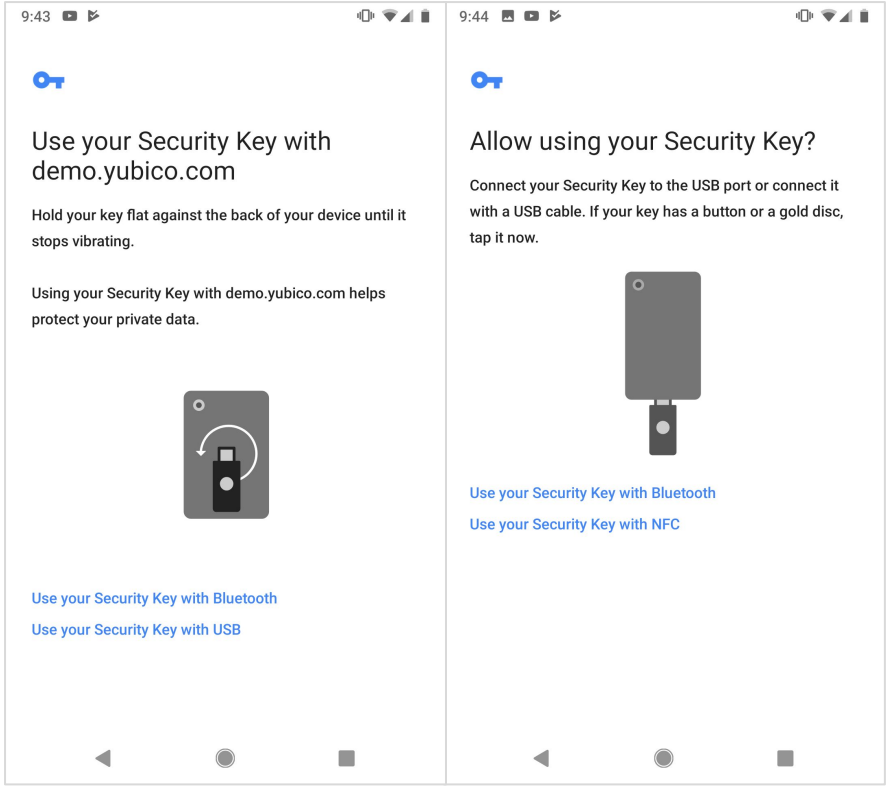
```
dictionary PublicKeyCredentialDescriptor {
    required PublicKeyCredentialType      type;
    required BufferSource                 id;
    sequence<AuthenticatorTransport>      transports;
};
```

```
enum AuthenticatorTransport {
    "usb",
    "nfc",
    "ble",
    "internal"
};
```

Cross-Platform Credential Request Example

PublicKeyCredentialRequestOptions

```
{
  "publicKey": {
    "allowCredentials": [
      {
        "id": "X9FrwMfmzj...",
        "type": "public-key",
        "transports": ["usb","nfc","ble"]
      }
    ],
    "challenge": "kYhXBWX0HO5GstIS02yPJVhiZ0jZLH7PpC4tzJI-ZcA=",
    "rpId": "demo.yubico.com",
    "timeout": 30000,
    "userVerification": "discouraged"
  }
}
```



Android - NFC

Android - USB

Cross-Platform Credential Request Example

PublicKeyCredentialRequestOptions

```
{
  "publicKey": {
    "allowCredentials": [
      {
        "id": "X9FrwMfmzj...",
        "type": "public-key",
        "transports": ["usb"]
      }
    ],
    "challenge": "kYhXBWX0HO5GstIS02yPJVhiZ0jZLH7PpC4tzJI-ZcA=",
    "rpId": "demo.yubico.com",
    "timeout": 30000,
    "userVerification": "discouraged"
  }
}
```

Windows - USB

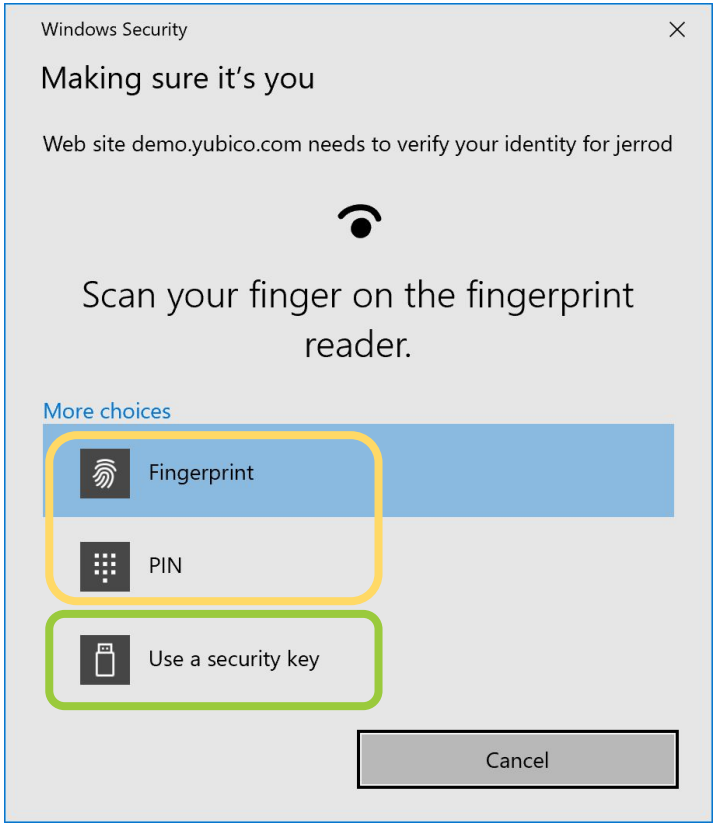
Cross-Platform + Platform Credential Request Example

PublicKeyCredentialRequestOptions

```
{
  "publicKey": {
    "allowCredentials": [
      {
        "id": "X9FrwMfmzj...",
        "type": "public-key",
        “transports”: [“usb”,”internal”]
      }
    ],
    "challenge": "kYhXBWX0HO5GstIS02yPJVhiZ0jZLH7PpC4tzJI-ZcA=",
    "rpId": "demo.yubico.com",
    "timeout": 30000,
    "userVerification": "discouraged"
  }
}
```

**Windows -
Hello Fingerprint/PIN**

**Windows -
USB**



FIDO2/WebAuthn – Learn More

Blogs

Microsoft Blog: All about FIDO2, CTAP2 and WebAuthn »

What is FIDO2? »

10 Things You've Been Wondering About FIDO2, WebAuthn, and a Passwordless World »

Passwordless Login comes to Microsoft Accounts »

FIDO2 Project »

Webinars

FIDO2 Authentication Demystified »

FIDO2 WebAuthn Data Flows, Attestation »

FIDO2 WebAuthn Server Validation »

FIDO2 Explained »

Resources

Web Authentication API - Web APIs | MDN »

FIDO2/WebAuthn Yubico Developer Program »

Yubico WebAuthn Demo Site »

FIDO Alliance WebAuth Overview »

FIDO Alliance WebAuthn Workshop »

Implementing WebAuthn

Open source libraries:

- Java: https://github.com/Yubico/java-webauthn-server
  - Full-featured implementation of all validation logic
  - Attestation validation, metadata
  - Includes example REST server
- Python: https://github.com/Yubico/python-fido2
  - Host library
  - Provides primitives for verifying signatures
- C: https://github.com/Yubico/libfido2
  - Host library
  - Provides primitives for verifying signatures

Specifications:

- W3C Web Authentication specification
- FIDO2 Client to Authentication Protocol

yubico

Trust the Net