

OpenSSH + FIDO workshop

Joost van Dijk | Yubico

Open Source Summit Europe - August 2025

About this workshop

- Exercises:
 - <https://github.com/YubicoLabs/fido-openssh-workshop>
- Basic OpenSSH knowledge assumed
- Some demos require a GitHub or GitLab account
- If you need one: grab a FIDO Security Key!

yubico

Before we start...

- If you haven't already, download/install:
 - OpenSSH client (version 8.2+, ideally 8.9+)
\$ ssh -V
 - Python 3.9+
\$ python3 -V
 - Docker Desktop (or something similar)
<https://www.docker.com/products/docker-desktop/>
 - An Ubuntu image (latest)
\$ docker pull ubuntu:latest
 - FIDO command-line tools
<https://developers.yubico.com/libfido2/>
\$ <pkg-mgr> install libfido2
(fido2-tools on debian/ubuntu)

Overview

- Intro - what is FIDO?
 - Tools: libfido2
- Hardware-backed SSH private keys
 - Exercise: sign in using hardware-backed keys
- Generating SSH signatures using security keys
 - Exercise: git commit signing
- Storing SSH certificates
 - Exercise: largeBlobs
- Device attestation and the FIDO metadata service
 - Exercise: generating attestations
- ssh-agent with security keys
 - Exercise: more secure agent-forwarding

Introduction

FIDO2 - concepts

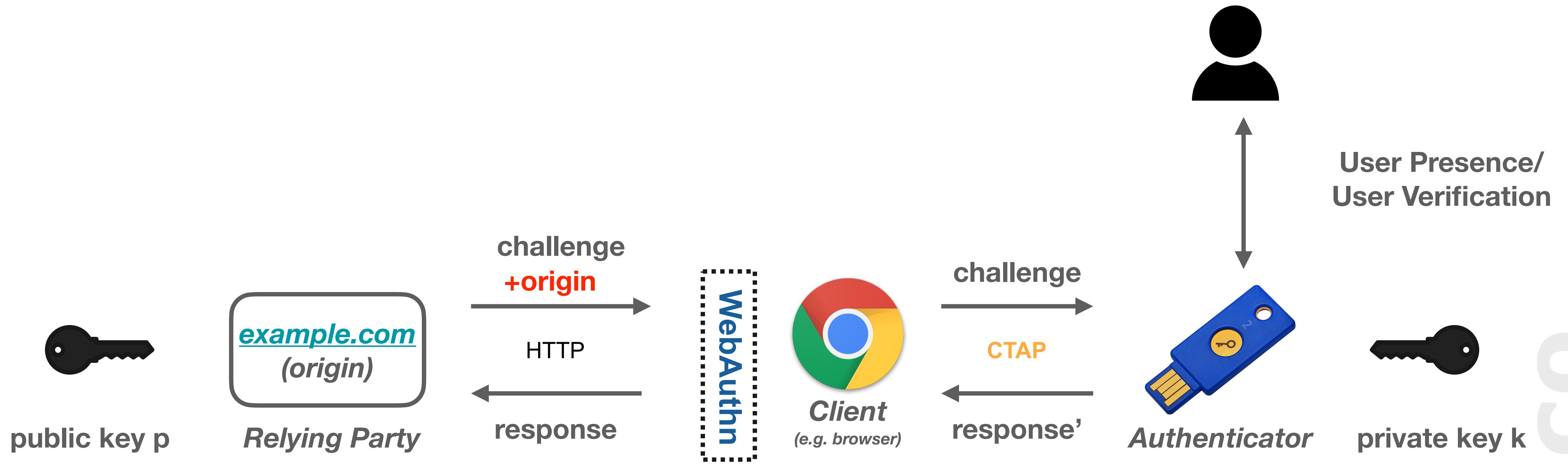
FIDO2

- Specifications:
 - CTAP - using a FIDO authenticator from a client (e.g. a browser)
 - Webauthn: API for using FIDO credentials in web applications

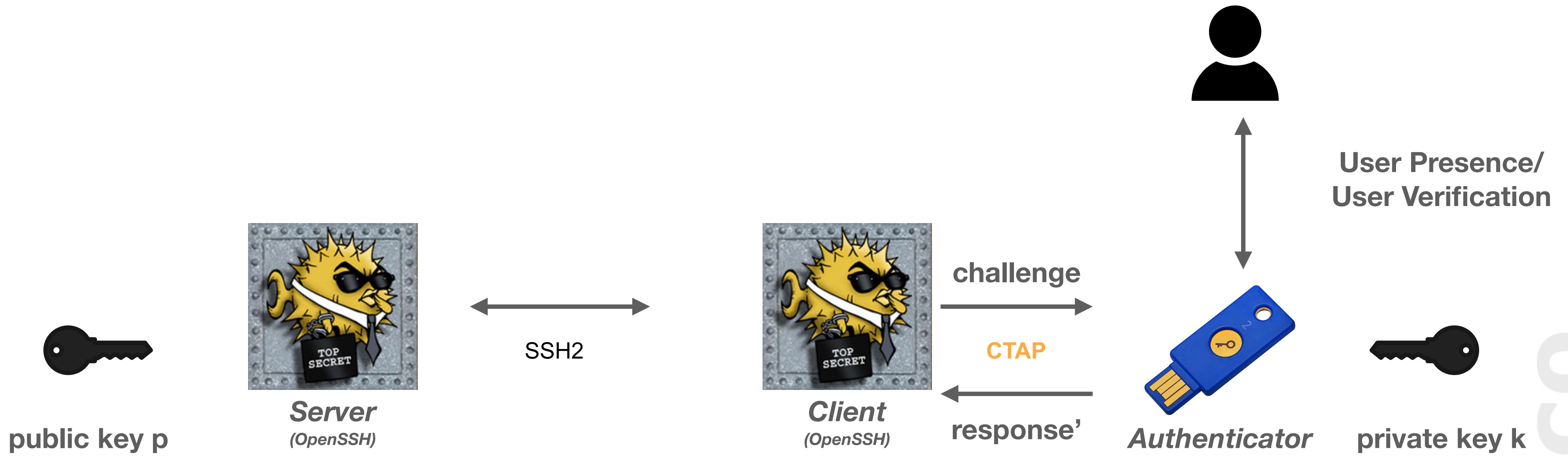
A screenshot of a web browser window displaying the FIDO Alliance Client to Authenticator Protocol (CTAP) specification. The title is "Client to Authenticator Protocol (CTAP)" in orange text. Below it, it says "Proposed Standard, June 15, 2021". The FIDO Alliance logo is in the top right. The URL in the address bar is "fidoalliance.org". The page includes links for "This version" (a blue link), "Previous Versions" (a blue link), and "Issue Tracking" (a blue link). A GitHub icon is in the bottom left corner.

A screenshot of a web browser window displaying the W3C Web Authentication: An API for accessing Public Key Credentials Level 2 recommendation. The title is "Web Authentication: An API for accessing Public Key Credentials Level 2" in blue text. Below it, it says "W3C Recommendation, 8 April 2021". The W3C logo is in the top right. The URL in the address bar is "w3.org". The page includes links for "This version" (a blue link) and "Latest published version" (a blue link). A GitHub icon is in the bottom left corner.

CTAP and Webauthn (simplified)

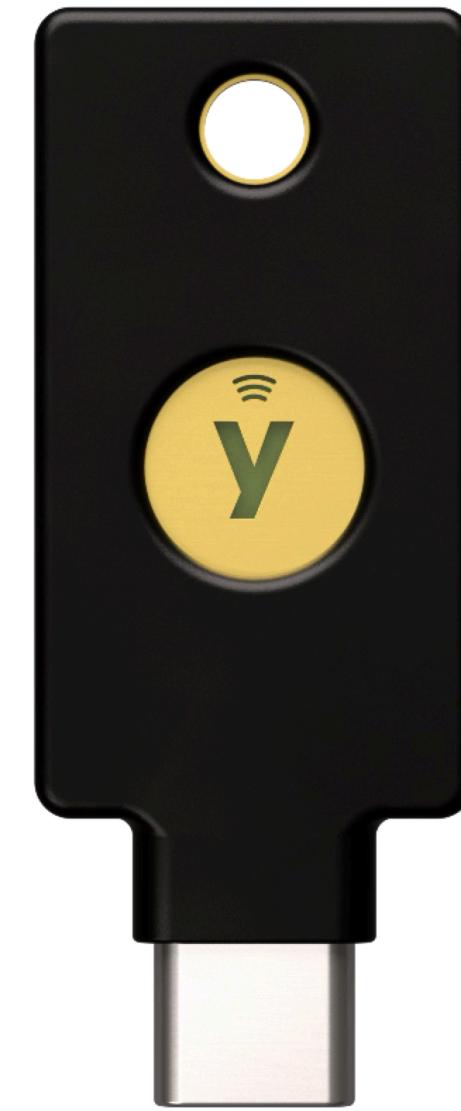


CTAP and SSH

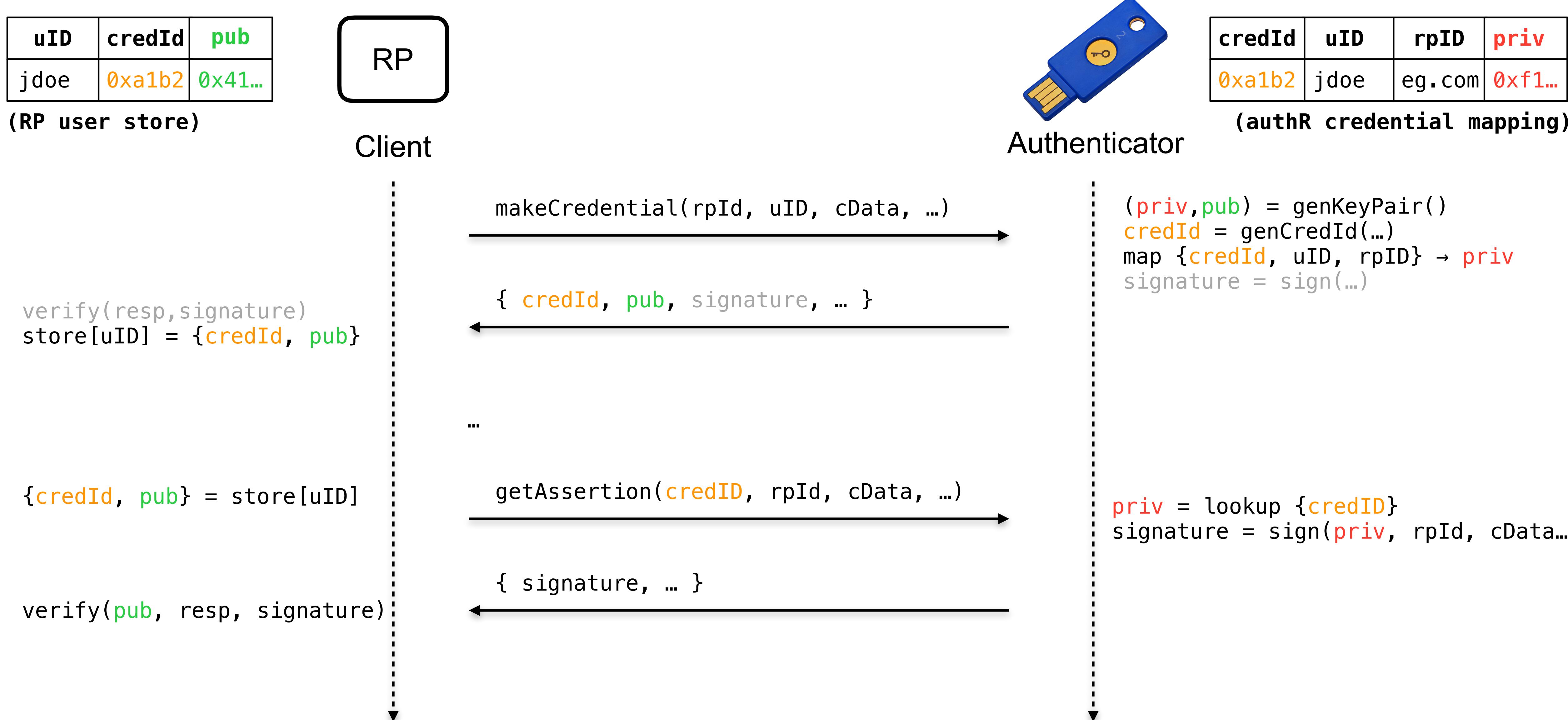


Why OpenSSH + FIDO?

- SSH Private Key
 - Generated and stored on secure hardware
(a FIDO Security Key)
 - Cannot be exported: a possession factor for MFA
- Before any private key operation
 - Option to require a button touch to prove **User Present**
 - Option to require PIN entry for **User Verification**
- Store associated SSH certificates on a portable device
- Cryptographically prove Security Key provenance (make and model)
- Simpler than using OpenPGP or PKCS#11



CTAP: operations (simplified)



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 FIDO credentials!

FIDO Security Keys: cross-platform authenticators

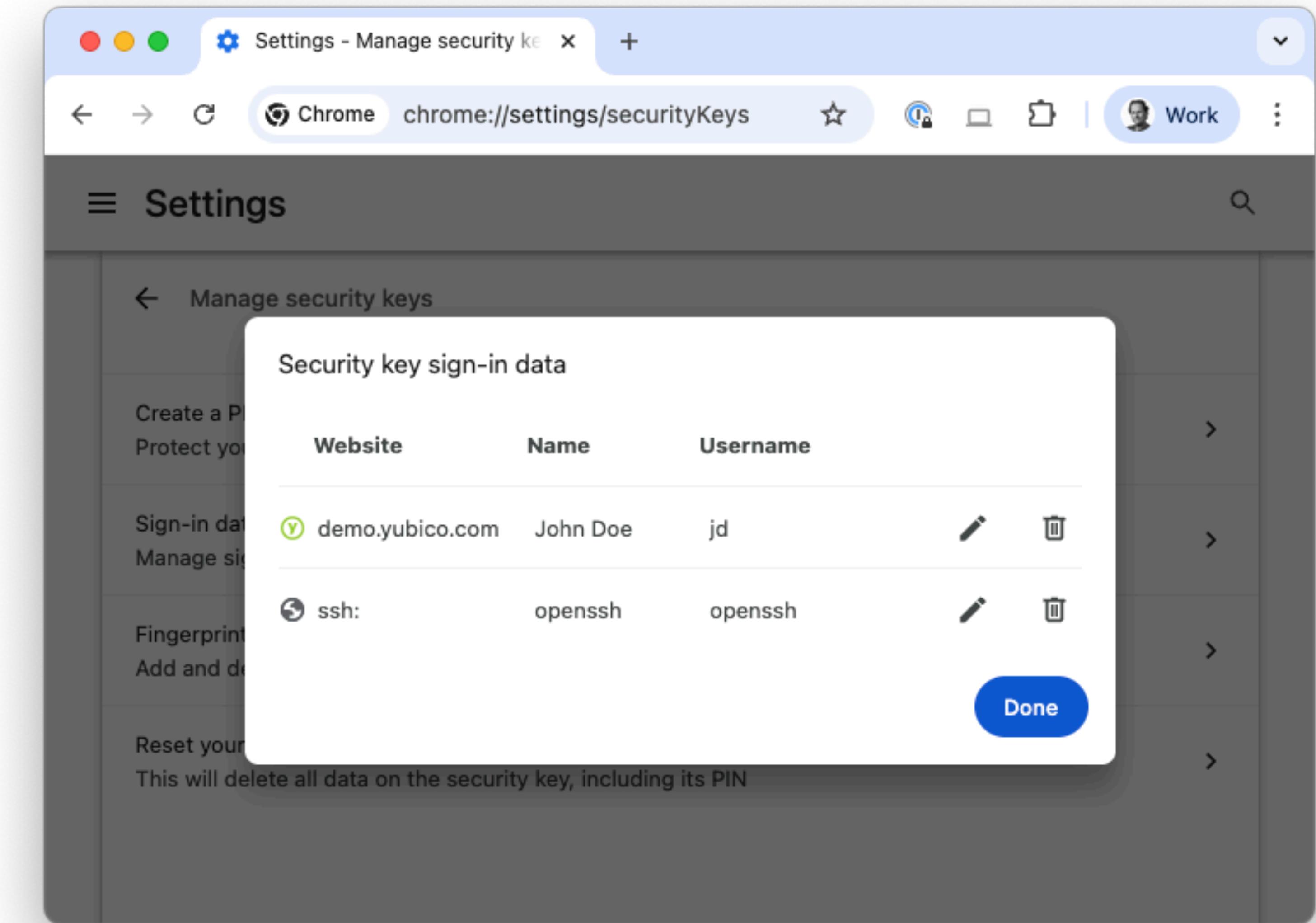


User Present vs User Verified

- User Present (UP): User is physically present
 - On a security key: a button touch
- User Verified (UV): User is verified, i.e. user authenticated to the Security Key
 - On a security key: entering a PIN
(on biometric models: touching a built-in fingerprint scanner)
 - On a platform authenticator: typically a biometric
(eg Touch ID/Face ID)

Resident vs non-resident Keys

- A Credential can be:
 - **Resident:** stored in a security key's internal memory
 - **Non-resident:** encrypted with a wrapping key and stored off-key
- Resident keys are called *Discoverable Credentials* since CTAP v2.1
- A security key can store a limited amount of resident keys



Passwordless sign-in with passkeys

A screenshot of a web browser window showing the GitHub Settings page for the account "dev-tunnel (dev-tunnel)". The left sidebar shows various settings categories like Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and licensing, Emails, Password and authentication (which is selected), Sessions, SSH and GPG keys, Organizations, Enterprises, and Moderation. The main content area is titled "Sign in methods" and lists four methods: Email, Password, Passkeys, and Google. The Passkeys section shows one passkey configured, associated with a YubiKey. The "Two-factor authentication" section below states that it is not enabled yet.

github.com/settings/security

Settings

dev-tunnel (dev-tunnel)
Your personal account

Public profile

Account

Appearance

Accessibility

Notifications

Access

Billing and licensing

Emails

Password and authentication

Sessions

SSH and GPG keys

Organizations

Enterprises

Moderation

Code, planning, and automation

Repositories

Codespaces

Models

Packages

Copilot

Pages

Saved replies

Sign in methods

Email
1 verified email configured

Manage

Password
Configured

Change password

Passkeys
1 passkey configured ^

Add passkey

My YubiKey Seen from this browser
Added on Sep 19, 2024 | Last used about 5 hours ago

Google
Sign in with your Google account

Connect

Two-factor authentication

Two-factor authentication is not enabled yet.

Two-factor authentication adds an additional layer of security to your account by requiring more than just a password to sign in.

Attestation

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

Cookies
Legal
Privacy
Terms of Use

Yubico © 2020. All Rights Reserved.

libfido2

- <https://github.com/Yubico/libfido2>
- Library and command-line tools
- Communicate with a FIDO device over USB or NFC
- Supports both FIDO U2F (CTAP 1) and FIDO2 (CTAP 2) protocols
- Works on Linux, macOS, Windows, OpenBSD, and FreeBSD
- Bindings for [.NET](#), [Go](#), [Perl](#), [Rust](#)
- Also available:
 - <https://github.com/Yubico/python-fido2>

Get Info

```
$ fido2-token -L
ioreg://4296874505: vendor=0x1050, product=0x0407 (Yubico YubiKey OTP+FIDO+CCID)
$ fido2-token -I ioreg://4296874505
proto: 0x02
major: 0x05
minor: 0x07
build: 0x04
caps: 0x05 (wink, cbor, msg)
version strings: U2F_V2, FIDO_2_0, FIDO_2_1_PRE, FIDO_2_1
extension strings: credProtect, hmac-secret, largeBlobKey, credBlob, minPinLength
transport strings: nfc, usb
algorithms: es256 (public-key), eddsa (public-key), es384 (public-key)
aaguid: d7781e5de35346aaafe23ca49f13332a
options: rk, up, noplus, noalwaysUv, credMgmt, authnrCfg, clientPin, largeBlobs, pinUvAuthToken, setMinPINLength, makeCredUvNotRqd
fwversion: 0x50704
maxmsgsz: 1536
maxcredcntlst: 8
maxcredlen: 128
maxcredblob: 32
maxlargeblob: 4096
maxrpids in minpinlen: 1
remaining rk(s): 98
minpinlen: 4
pin protocols: 2, 1
pin retries: 8
pin change required: false
uv retries: undefined
```

Resources

- Developer Program: <https://dev.yubi.co/>
- CTAP 2.1 spec: <https://fidoalliance.org/specs/fido-v2.2-ps-20250228/>
- FIDO dev forum: <https://groups.google.com/a/fidoalliance.org/g/fido-dev>
- libfido2: <https://github.com/Yubico/libfido2>
- FIDO Metadata:
 - <https://fidoalliance.org/metadata/>
 - <https://opotonnier.github.io/fido-mds-explorer/>
- SSH support for security keys:
 - <https://github.com/openssh/openssh-portable/blob/master/PROTOCOL.u2f>

yubico

Questions?