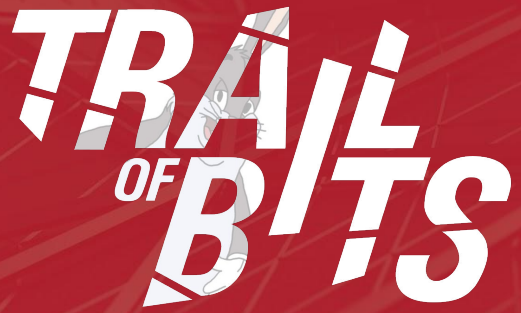



# Improving PyPI's security with Two Factor Authentication

William Woodruff

# Hi

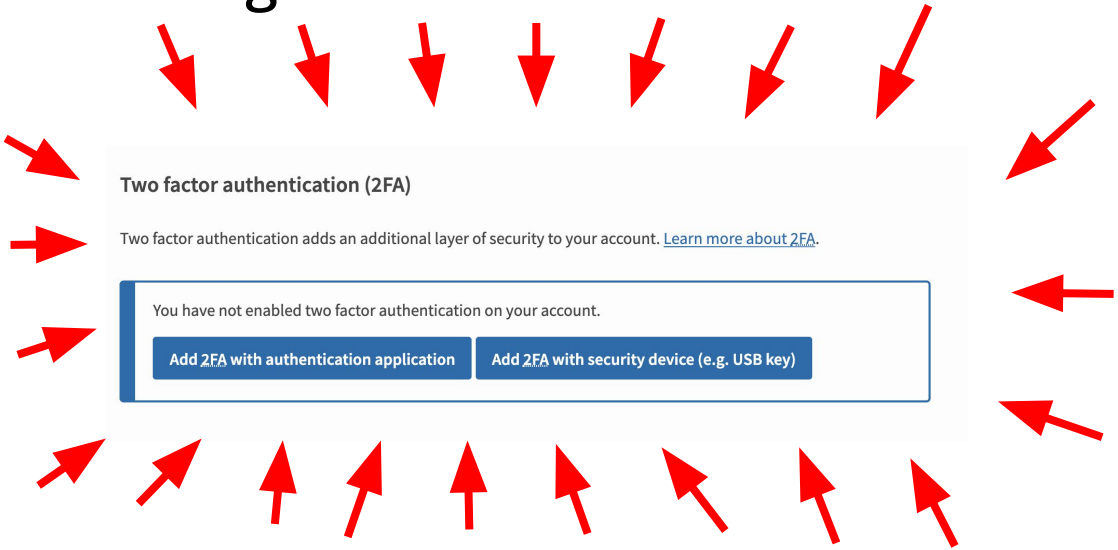
- **William Woodruff (@8x5clPW2)**
  - Security Engineer (R&D) at Trail of Bits
  - Research: program analysis
  - Engineering: security oriented client work, mostly (F)OSS
- **Agenda**
  - You enable 2FA on your PyPI account
  - We talk about different kinds of 2FA & why/how to use them
  - We talk about PyPI's codebase and how awesome it is

# Some thank-yous are in order

- **This talk is about 2FA, but a lot of other things have been added to PyPI over the last 6 months:**
  - API tokens, WCAG accessibility, internationalization, audit logs, …
- **None of this would have been possible without:**
  - Sumanah Harihareswara (brainwane)
  - Nicole Harris (nlhkabu)
  - Ernest W. Durbin (ewdurbin)
  - Dustin Ingram (di)
  - Donald Stufft (dstufft)

# Newsflash!!!

- PyPI supports two factor authentication
- Enable it!! Right now!!!! During the talk!!!!!!

# Make these numbers better!!!

- **9/24:**
  - ~375,000 total users
  - ~3000 accounts with TOTP enabled (< 1%)
  - ~300 with WebAuthn (< 0.1%)
  - 😢 (~98.9%)
- **Silver lining:**
  - 12% of top 100 packages have 100% 2FA
  - ❤️ PyOpenSSL maintainers (5/5!)

# Two factor authentication: a primer

- **Authentication: proving to a service that you are who you say you are (a superstar, so have no fear)**
  - Discrete from authorization (proving that you have rights to a resource)
- **First factor: your password, or something else you know**
  - Proves knowledge of some secret
- **Second factor: something that you (and only you) have**
  - Proves physical possession of some (hopefully) hard-to-steal token

**For this talk, 2FA = MFA!!!**

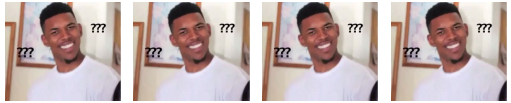
# 2FA: why even?

- **Adding a second factor {prevents, mitigates}:**
  - Credential stuffing
  - Most phishing attacks
  - Password/DB leaks*
  - Your partner creeping on your DMs

- **2FA isn't a panacea**
  - Typosquatting is common in language PMs (PyPI has protections)
  - Malicious/negligent owners can always sell their package

*** Unless the DB leak also includes TOTP secrets**

# A whirlwind tour of 2FA methods

- **SMS/voice**
  - Do not add to a new 2FA implementation
- **HOTP-based physical keys**
  - 
- **TOTP**
  - Very okay
- **WebAuthn (and also U2F)**
  - The new hotness

# SMS/voice 2FA: Just don't*

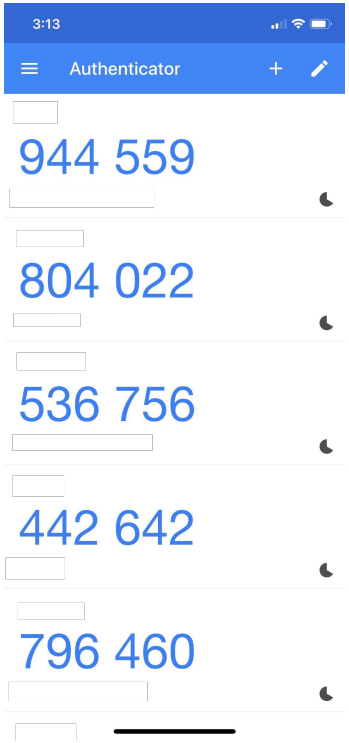
- **User provides phone number during registration**
  - During authentication, service contacts the user with a one time code
  - User enters code as their second factor
  - Proof: User has that phone number
- **Problem: Nothing about the phone network is secure**
  - SS7 attacks, SIM ports & jacking, SMS interception
- **Do not add SMS to a new 2FA implementation in 2019!**

# HOTP-like tokens

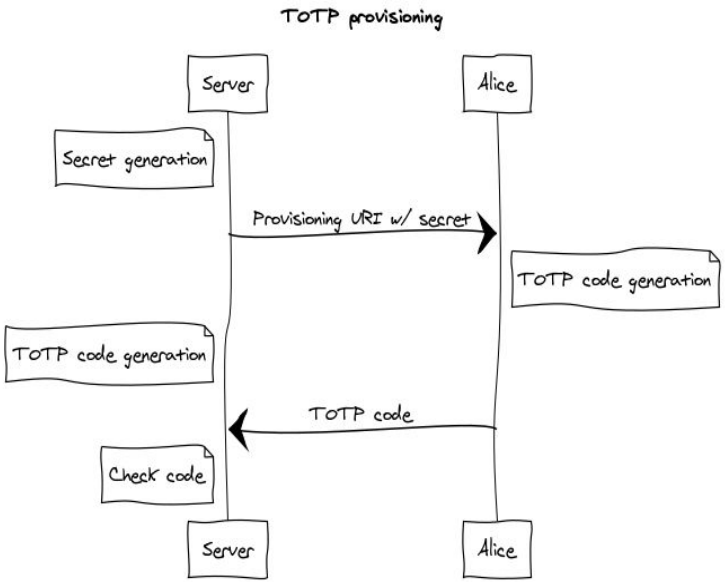

- **Weird proprietary solutions like SecurID**
  - "HOTP-like" b/c who knows what the hell they're doing
  - Lots of different companies, devices
- **User is registered, is given a physical token**
  - Physical token gives the user a code to enter
  - Proof: only the user's physical token can generate the right code
    - ...presumably
    - Many of these companies also sell smart cards

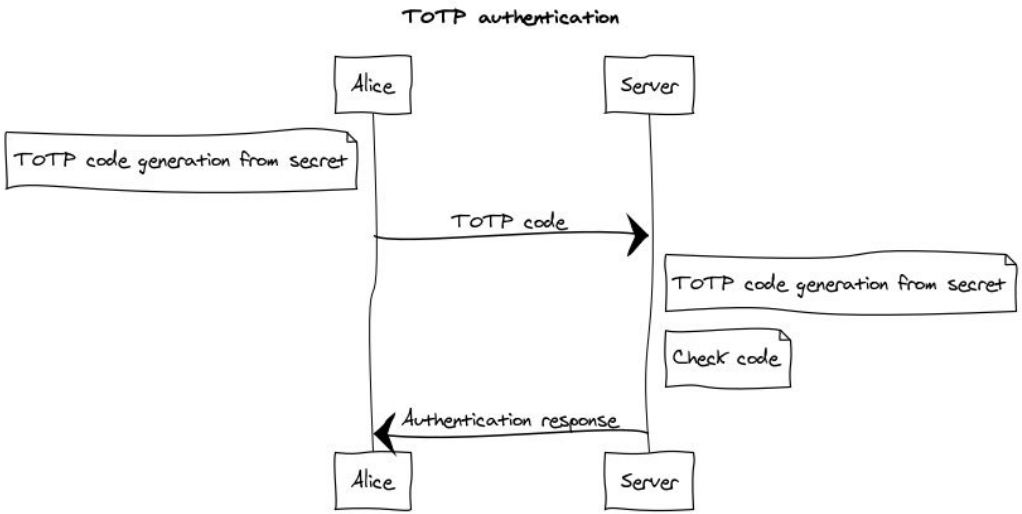
# TOTP: Your baseline 2FA

- **“Time-based one-time passwords”**
- **Symmetric cryptography w/ a single shared secret**
  - Server generates and shares secret with client
  - Client and server craft codes from secret, server verifies codes
  - Moving factor? The current time, window boxed
- **RFC: 6238**
  - Based on HOTP (4226), you can read + understand both in a day!



# TOTP from 1000 feet



# TOTP dos and don’ts

- **Do:**
  - Require users to enter a TOTP code to confirm enrollment
  - Provide both a provisioning URI **and** a QR code
    - Embed the user’s username + a sensible issuer name!
- **Don’t:**
  - Use uncommon TOTP parameters (bad client support)
    - Just use 6-digit codes, 30s windows (+2 for drift), SHA-1
  - Nag users for TOTP codes on every single action

# WebAuthn: *you know we stan*

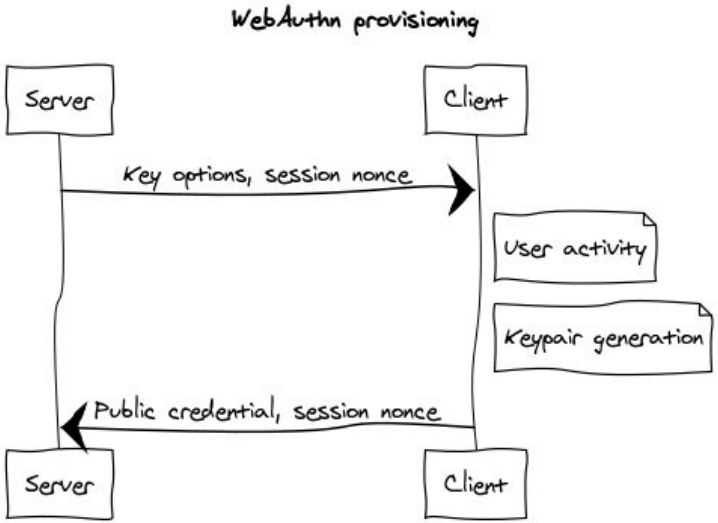
- **Terminological spaghetti: U2F, FIDO2, CTAP1, CTAP2**
  - WebAuthn covers all; users still treat U2F as a generic term :(

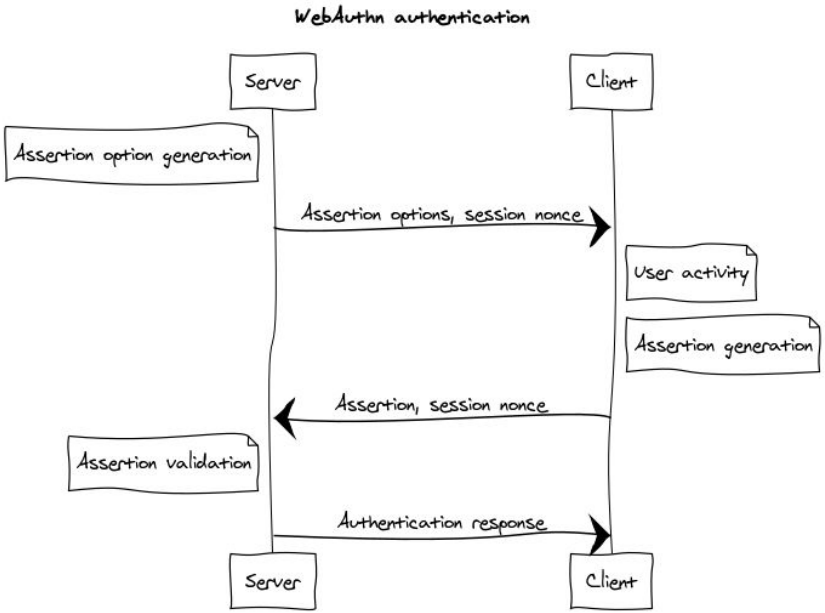

- **Asymmetric, public-key cryptography**
  - Client and server negotiate key options, client generates keypair
  - Client shares public key, generates assertions from private key


- **Hardware agnostic (as long as it speaks the protocol)**
  - USB, bluetooth, NFC, fingerprint, soft tokens are all available

- **W3C standard!!! Widespread browser support!!!!!!!**

# WebAuthn from 1000 feet



# WebAuthn dos and don’ts

- **Do:**
  - Allow anonymous attestation (ECDAA): TouchID can do WebAuthn!
  - Support multiple WebAuthn keys per user
  - Force users to tag/label their keys
- **Don’t:**
  - Make users choose between TOTP and WebAuthn; let them do both!
  - Use u2f.js or any other pre-WebAuthn library (poor browser support)

# 2FA: A technical wrap-up

- **You should be adding 2FA to your sites/services**
  - If you already have SMS 2FA, you should be phasing it out
- **TOTP is easy to add and is the baseline**
  - pyca/cryptography provides a high-quality, misuse-resistant impl.
- **WebAuthn is easy-ish to add and is Good™**
  - Duo's py_webauthn is a little rough, but production-ready
  - Migrating from U2F will improve your browser support
    - death to u2f.js

# Let's talk about PyPI

- **Legacy codebase**
  - I don't know anything about this
  - Probably haunted
- **Warehouse (2015? — present)**
  - very nice 
  - 100% branch coverage in unit tests
  - Idiomatic ~~MVC~~ RV usage of Pyramid
  - Easy to deploy locally (a few `make` targets)

# Adding 2FA to PyPI

- **Phase 1: TOTP**
  - pypa/warehouse#5567
  - Began 3/13, merged 5/4
  - +1,673 −69
- **Phase 2: WebAuthn**
  - pypa/warehouse#5795
  - Began 5/6, merged 7/17
  - +2,383 −122

Mostly unit tests!!!

# Adding 2FA to PyPI: fun bugs

- **Historical behavior: successful password reset also logged user in**
  - Result: 2FA bypass!
- **Initial TOTP impl allowed code reuse within the window**
  - Result: MiTB/phishing weakness!
- **Inconsistent WebAuth implementations in browsers!**
  - Not all browsers require secure contexts for all WebAuthn URLs!!!
  - Fetch API inconsistencies w/ Request.credentials!!!

# Learn from our ~~mistakes~~ successes

- There are relatively few non-demo, production-quality WebAuthn implementations available for reference
- PyPI is now one of them
- It's well documented
- It's well tested
- It's loved by children and parents alike

# Thank you!


**William Woodruff**

Security Engineer

william@trailofbits.com

www.trailofbits.com

# References

Getting 2FA Right in 2019

https://blog.trailofbits.com/2019/06/20/getting-2fa-right-in-2019/

How effective is basic account hygiene at preventing hijacking?

https://security.googleblog.com/2019/05/new-research-how-effective-is-basic.html

WebAuthn W3C TR

https://www.w3.org/TR/webauthn/

PyPI Now Supports Two Factor Login via WebAuthn

https://pyfound.blogspot.com/2019/06/pypi-now-supports-two-factor-login-via.html

Use two-factor auth to improve your PyPI account's security

https://pyfound.blogspot.com/2019/05/use-two-factor-auth-to-improve-your.html

TRAIL OF BITS