

2550 Intro to cybersecurity L4

abhi shelat

Passwords, recap from L3

Password Storage Summary

- 1. Never store passwords in plain text**
 - 2. Always salt and hash passwords before storing them**
 - 3. Use hash functions with a high work factor**
 - 4. Implement honeywords to detect breaches**
- These rules apply to any system that needs to authenticate users
 - Operating systems, websites, etc.

Still one problem?

The image shows a Google login interface. At the top is the Google logo. Below it, the text "Hi abhi" is displayed. Underneath is a dropdown menu showing a profile picture and the email address "hi.abhi@gmail.com" with a downward arrow. Below this is a password input field with the placeholder text "Enter your password". The field contains two dots and has icons for password strength and visibility on the right. Below the password field are two links: "Forgot password?" and a blue "Next" button. At the bottom of the page, there is a language selector "English (United States)" with a dropdown arrow, and links for "Help", "Privacy", and "Terms".

Google

Hi abhi

hi.abhi@gmail.com ▼

Enter your password

..

[Forgot password?](#) [Next](#)

English (United States) ▼ [Help](#) [Privacy](#) [Terms](#)

Password Recovery/Reset

- Problem: hashed passwords cannot be recovered (hopefully)



“Hi... I forgot my password. Can you email me a copy? Kthxbye”

- This is why systems typically implement password **reset**
 - Use out-of-band info to authenticate the user
 - Overwrite `hash(old_pw)` with `hash(new_pw)`
- This step introduces a calculated vulnerability.

Other roots of identity



Account recovery

 hi.abhi@gmail.com ▾

Enter the last password you remember using with this Google Account

Enter last password

[Try another way](#)

Next

English (United States) ▾

[Help](#)

[Privacy](#)

[Terms](#)

Forgot username or password

Identification

[Have a question? >](#)

Help us verify your identity.

For your security, please choose one of the options to verify your identity and provide the other requested information.

Choose one

Social Security number ▾

Social Security number

xxx-xx-xxxx 

Don't have a Social Security number? >

Account type

☒ Chase ATM/debit/prepaid card or credit card



☐ Chase commercial loan

☐ Other Chase account (e.g., checking, savings, mortgage application, commercial term loan, auto loan or lease)

Choosing Passwords

Bad Algorithms

Better Heuristics

Password Reuse

Two Factor Authentication

Biometrics

SMS

Authentication Codes

Smartcards & Hardware Tokens

Types of Secrets

- Actors provide their secret to **log-in** to a system
- Three classes of secrets:
 1. Something you know
 - Example: a password
 2. Something you have
 - Examples: a smart card or smart phone
 3. Something you are
 - Examples: fingerprint, voice scan, iris scan

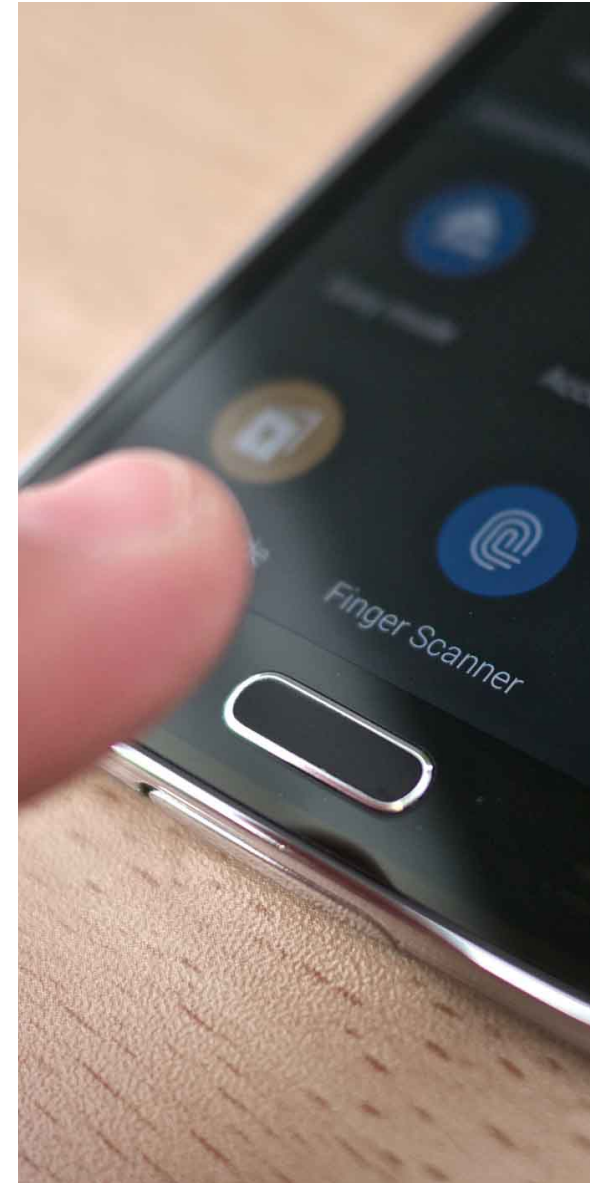
Biometrics

- ancient Greek: bios = "life", metron = "measure"
- Physical features
 - Fingerprints
 - Face recognition
 - Retinal and iris scans
 - Hand geometry
- Behavioral characteristics
 - Handwriting recognition
 - Voice recognition
 - Typing cadence
 - Gait

Fingerprints

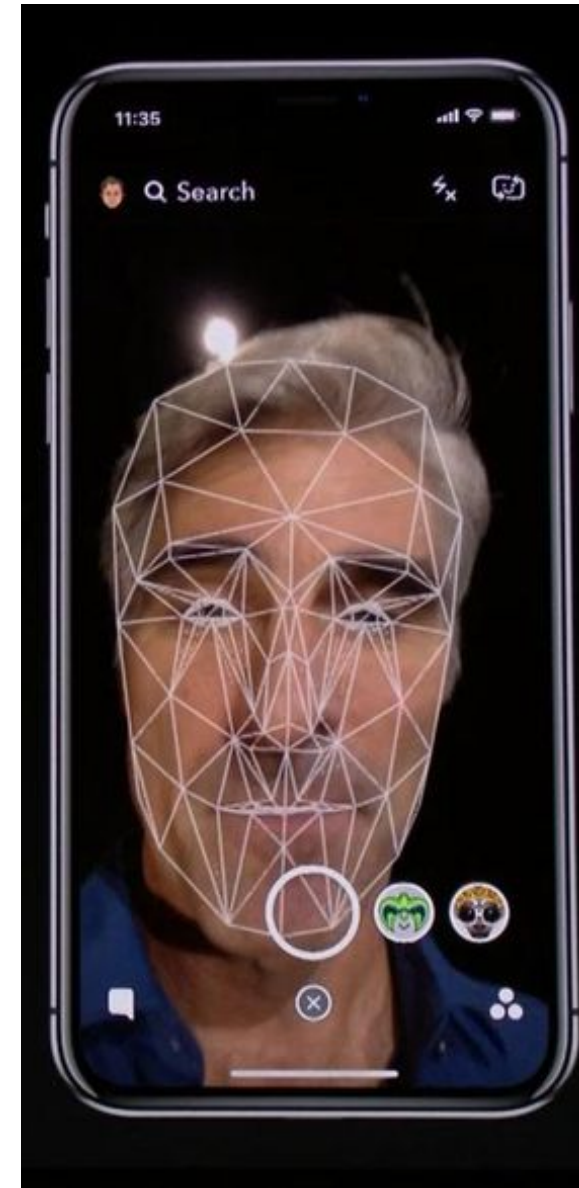
- Ubiquitous on modern smartphones, some laptops
- Secure?
 - May be subpoenaed by law enforcement
 - Relatively easy to compromise
 1. Pick up a latent fingerprint (e.g. off a glass) using tape or glue
 2. Photograph and enhance the fingerprint
 3. Etch the print into gelatin backed by a conductor
 4. Profit ;)

https://www.theregister.co.uk/2002/05/16/gummi_bears_defeat_fingerprint_sensors/



Facial Recognition

- Popularized by FaceID on the iPhone X
- Secure?
 - It depends
- Vulnerable to law enforcement requests
- Using 2D images?
 - Not secure
 - Trivial to break with a photo of the target's face
- Using 2D images + 3D depth maps?
 - More secure, but not perfect
 - Can be broken by crafting a lifelike mask of the target



Fundamental Issue With Biometrics

- Biometrics are immutable
 - You are the password, and you can't change
 - Unless you plan on undergoing plastic surgery?
- Once compromised, **there is no reset**
 - Passwords and tokens can be changed
- Example: the Office of Personnel Management (OPM) breach
 - US gov agency responsible for background checks
 - Had fingerprint records of all people with security clearance
 - Breached by China in 2015, all records stolen :(

Something You Have

- Two-factor authentication has become more commonplace
- Possible second factors:
 - SMS passcodes
 - Time-based one time passwords
 - Hardware tokens

SMS Two Factor

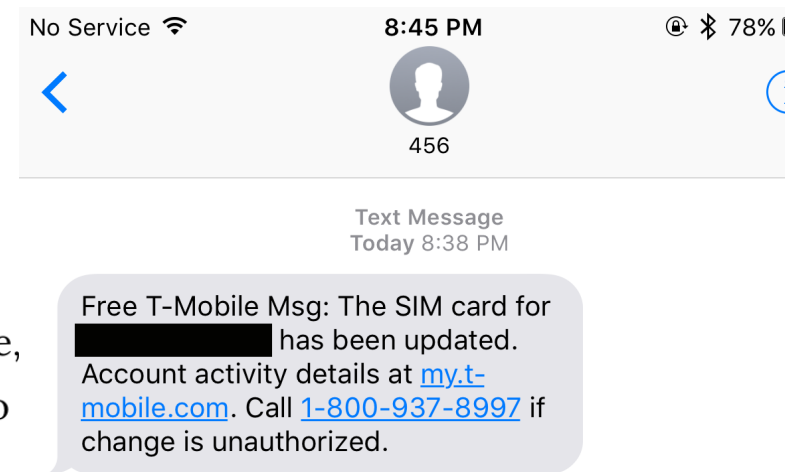
- Relies on your phone number as the second factor
 - Key assumption: only your phone should receive SMS sent to your number
- SMS two factor is deprecated. Why?
- Social engineering the phone company
 1. Call and pretend to be the victim
 2. Say “I got a new SIM, please activate it”
 3. If successful, phone calls and SMS are now sent to your SIM in your phone, instead of the victim
- Not hypothetical: successfully used against many victims



First, criminals call a cell phone carrier's tech support number pretending to be their target. They explain to the company's employee that they "lost" their SIM card, requesting their phone number be transferred, or ported, to a new SIM card that the hackers themselves already own. With a bit of social engineering—perhaps by providing the victim's Social Security Number or home address (which is often available from one of the many data breaches that have happened in the last few years)—the criminals convince the employee that they really are who they claim to be, at which point the employee ports the phone number to the new SIM card.

Game over.

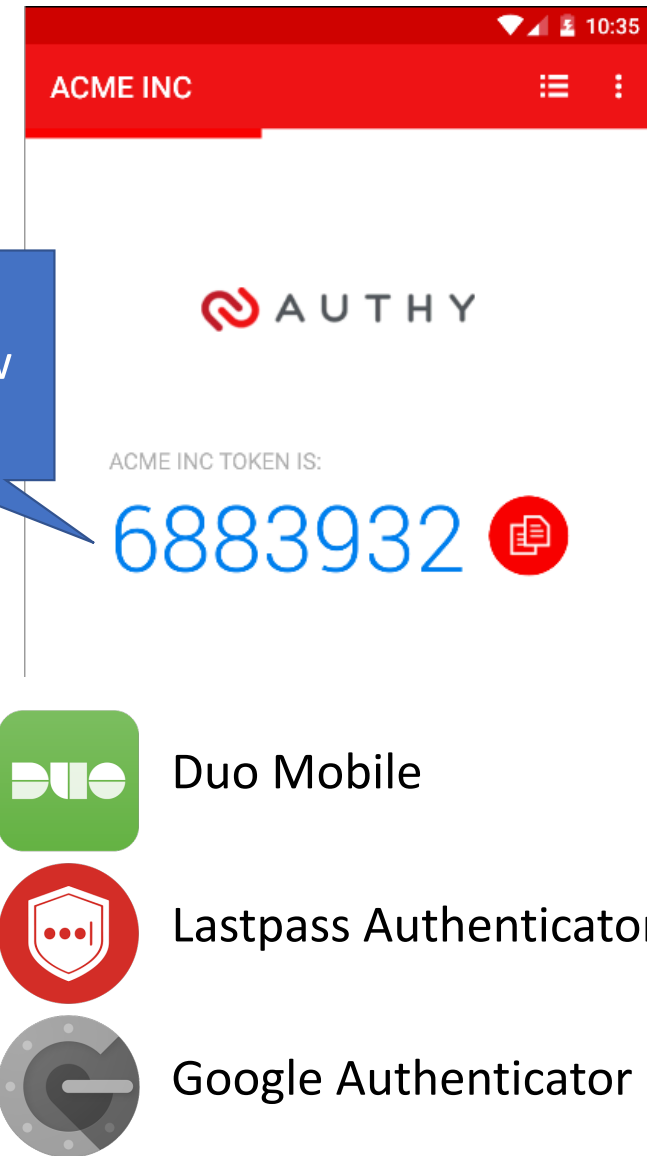
"With someone's phone number," a hacker who does SIM swapping told me, "you can get into every account they own within minutes and they can't do anything about it."



One Time Passwords

- Generate ephemeral passcodes that change over time
- To login, supply normal password and the current one time password
- Relies on a shared secret between your mobile device and the service provider
- Shared secret allows both parties to know the current one time password

Changes
every few
minutes



Time-based One-time Password Algorithm

$T0$ = <the beginning of time, typically Thursday, 1 January 1970 UTC>

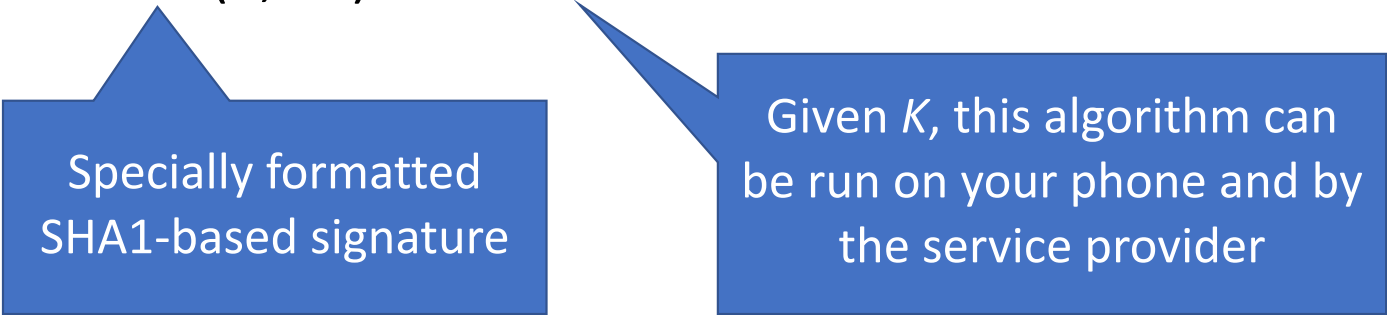
Tl = <length of time the password should be valid>

K = <shared secret key>

d = <the desired number of digits in the password>

$TC = \text{floor}((\text{unixtime}(\text{now}) - \text{unixtime}(T0)) / Tl),$

$\text{TOTP} = \text{HMAC}(K, TC) \% 10^d$



Specially formatted
SHA1-based signature

Given K , this algorithm can
be run on your phone and by
the service provider

Secret Sharing for TOTP

Enable Two-Step Sign in

An authenticator app generates the code automatically on your smartphone. Free apps are available for all smartphone platforms including iOS, Android, Blackberry and Windows. Look for an app that supports time-based one-time passwords (TOTP) such as Google Authenticator or Duo Mobile.

To set up your mobile app, add a new service and scan the QR code.



If you can't scan the code, enter this secret key manually: fvxo [blurred]

[USE SMS INSTEAD](#)[CANCEL](#)[NEXT STEP](#)

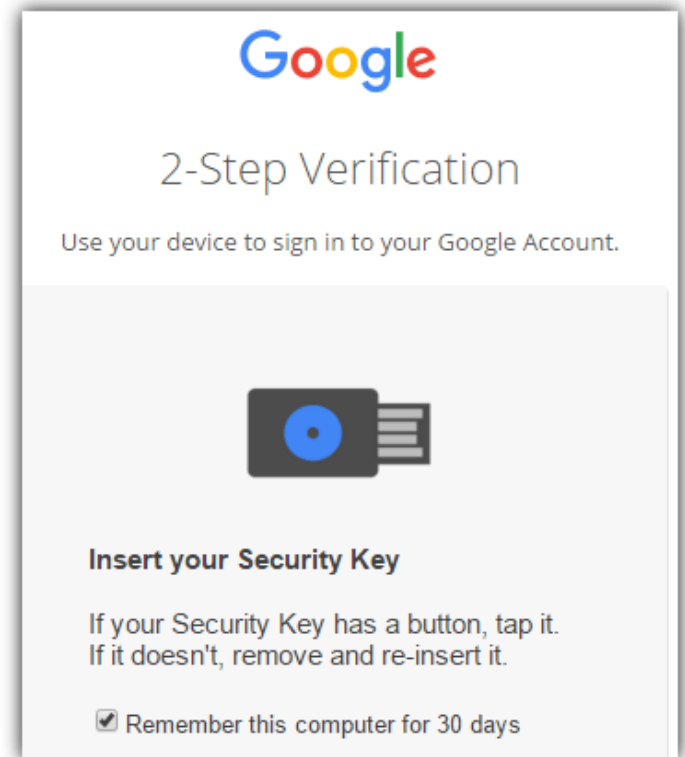
Hardware Two Factor

- Special hardware designed to hold cryptographic keys
- Physically resistant to key extraction attacks
 - E.g. scanning tunneling electron microscopes
- Uses:
 - 2nd factor for OS log-on
 - 2nd factor for some online services
 - Storage of PGP and SSH keys



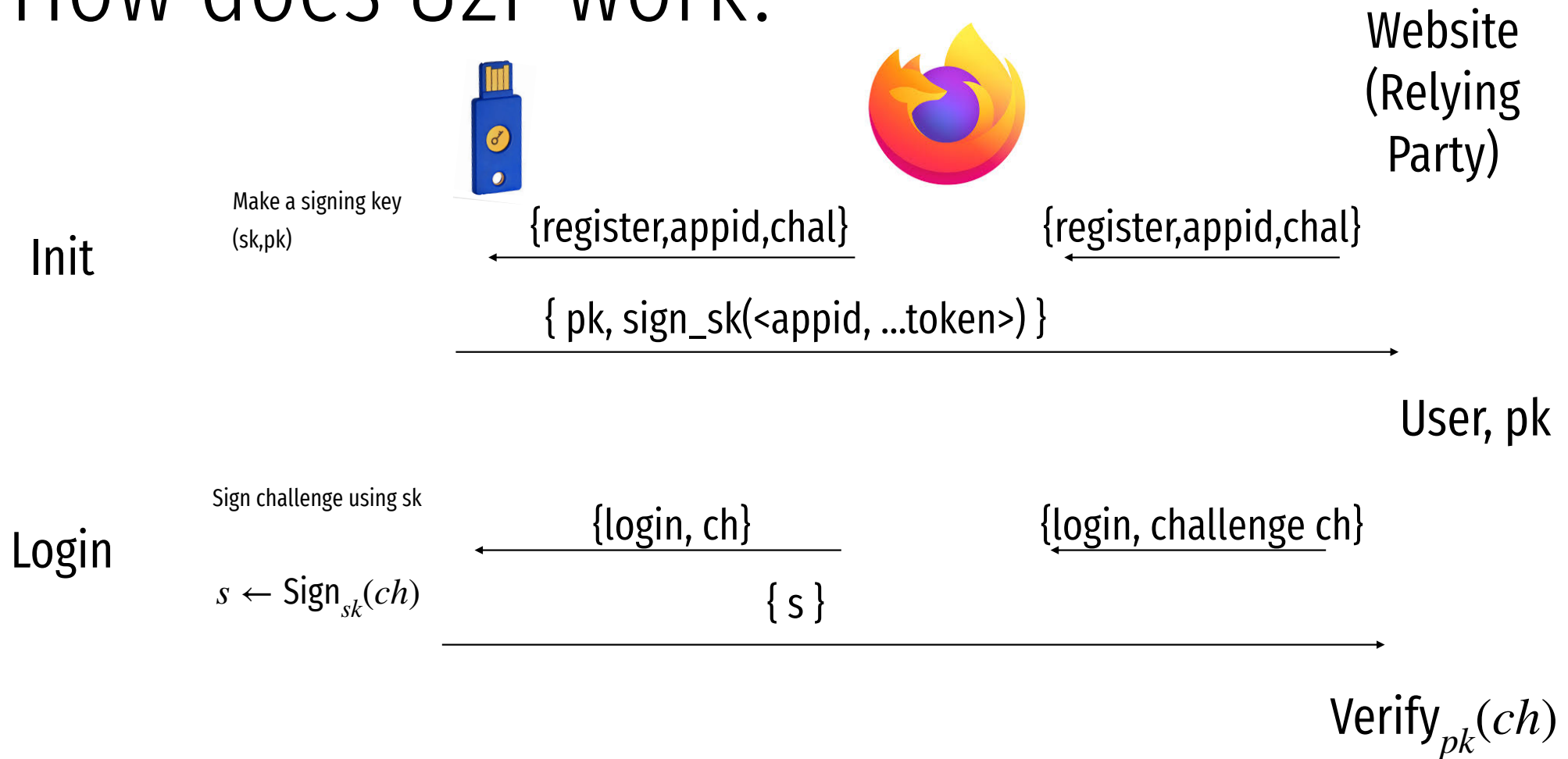
Universal 2nd Factor (U2F)

- Supported by Chrome, Opera, and Firefox (must be manually enabled)
- Works with Google, Dropbox, Facebook, Github, Gitlab, etc.
- Pro tip: always buy 2 security keys
 - Associate both with your accounts
 - Keep one locked in a safe, in case you lose your primary key ;)



How does U2F work?

Note: This version can be improved.



Vulnerable to simple attack



Welcome

 hi.abhi@gmail.com ▼

Enter your password  

[Forgot password?](#)

Next

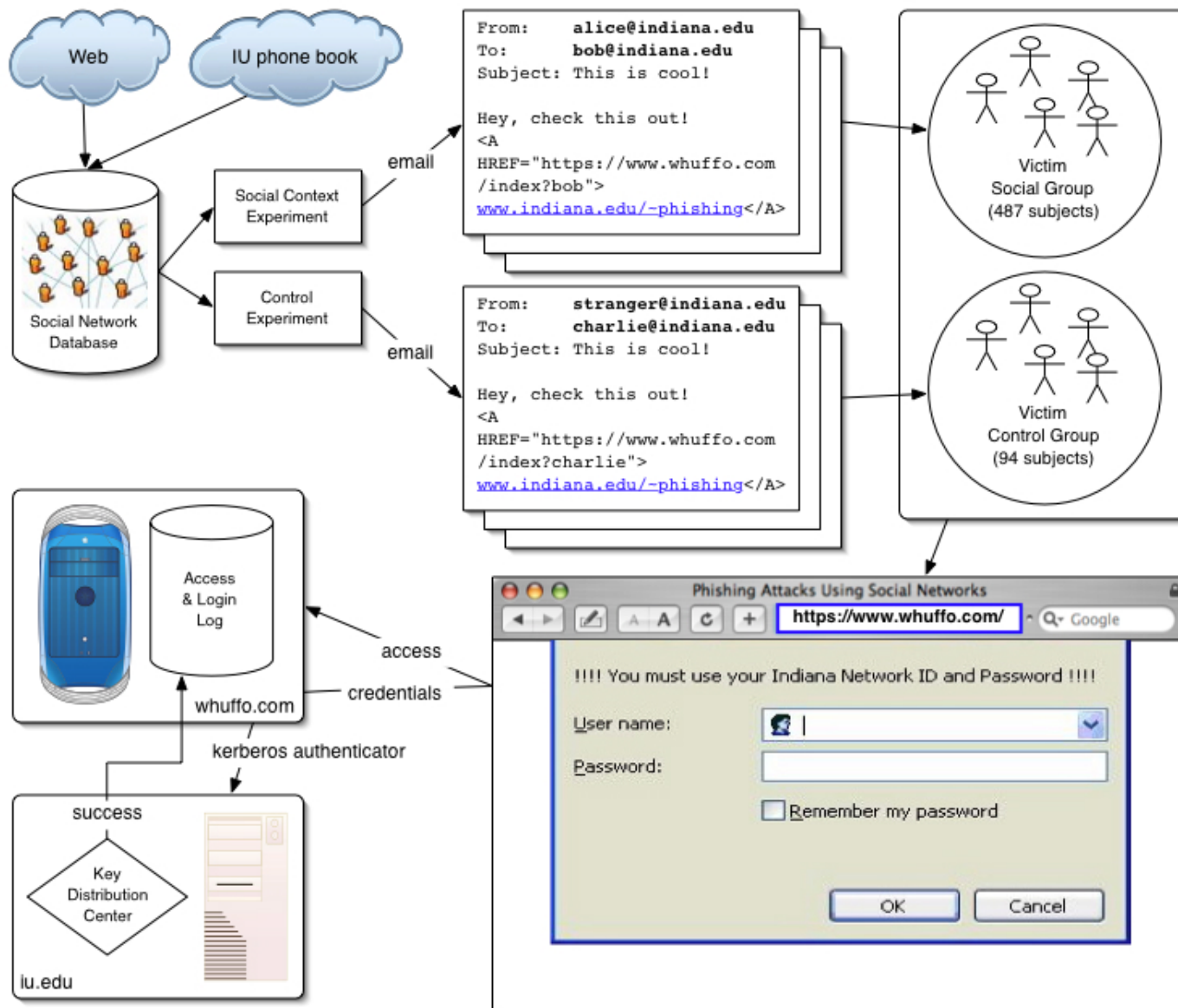
Simple Phishing

Lure: A spammed email with a call to action from a seemingly legitimate source encouraging the user to visit a hook website.

Hook: A website designed to mimic legitimate site and collect confidential information.

Spear Phishing @ IU

Experiment by T. Jagatic, N. Johnson, M. Jakobsson, F. Menczer.



Control Phishing Success Rate:

9-23%

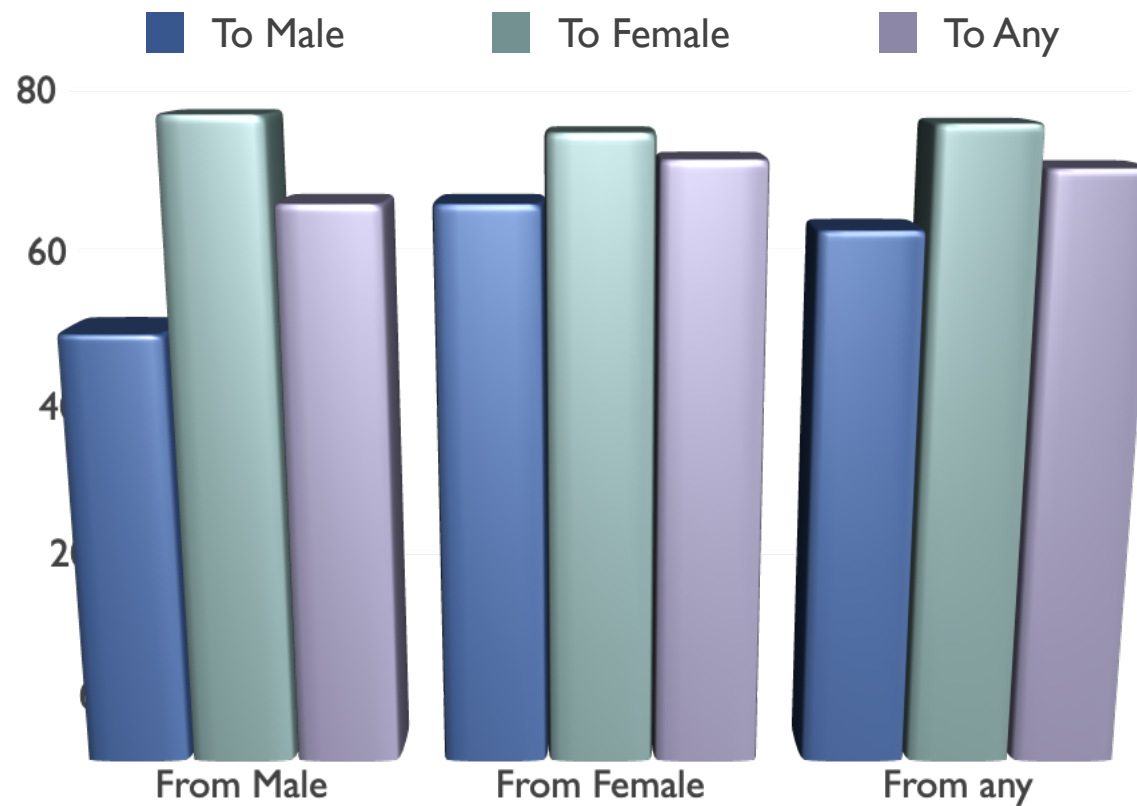
with 95% Confidence Interval

Spear Phishing Success Rate:

68-72%

with 95% Confidence Interval

Spear Phishing Success Rate by Gender



VOIP Phishing

Lure: Get victim to call a bogus 800... number about their account.

Hook: Have the human on the other end extract the victim's information.

From: FlagStar Bank <usflag60536@flagstar.com>
Date: 11 Sep 2007 10:55:21 -0400
To: <samyers@indiana.edu>
Subject: You have one new private message

Dear FlagStar Bank card holder,

You have one new private message.

Please call free 800-870-8124 to listen to your private message.

Copyright ©2007 FlagStar Bank

Source: Steven Myers, IU



Someone has your password

Hi William

Someone just used your password to try to sign in to your Google Account

Details:

Tuesday, 22 March, 14:9:25 UTC

IP Address: 134.249.139.239

Location: Ukraine

Google stopped this sign-in attempt. You should change your password immediately.

[CHANGE PASSWORD](#)

Best,
The Gmail Team

You received this mandatory email service announcement to update you about important changes to your Google product or account.

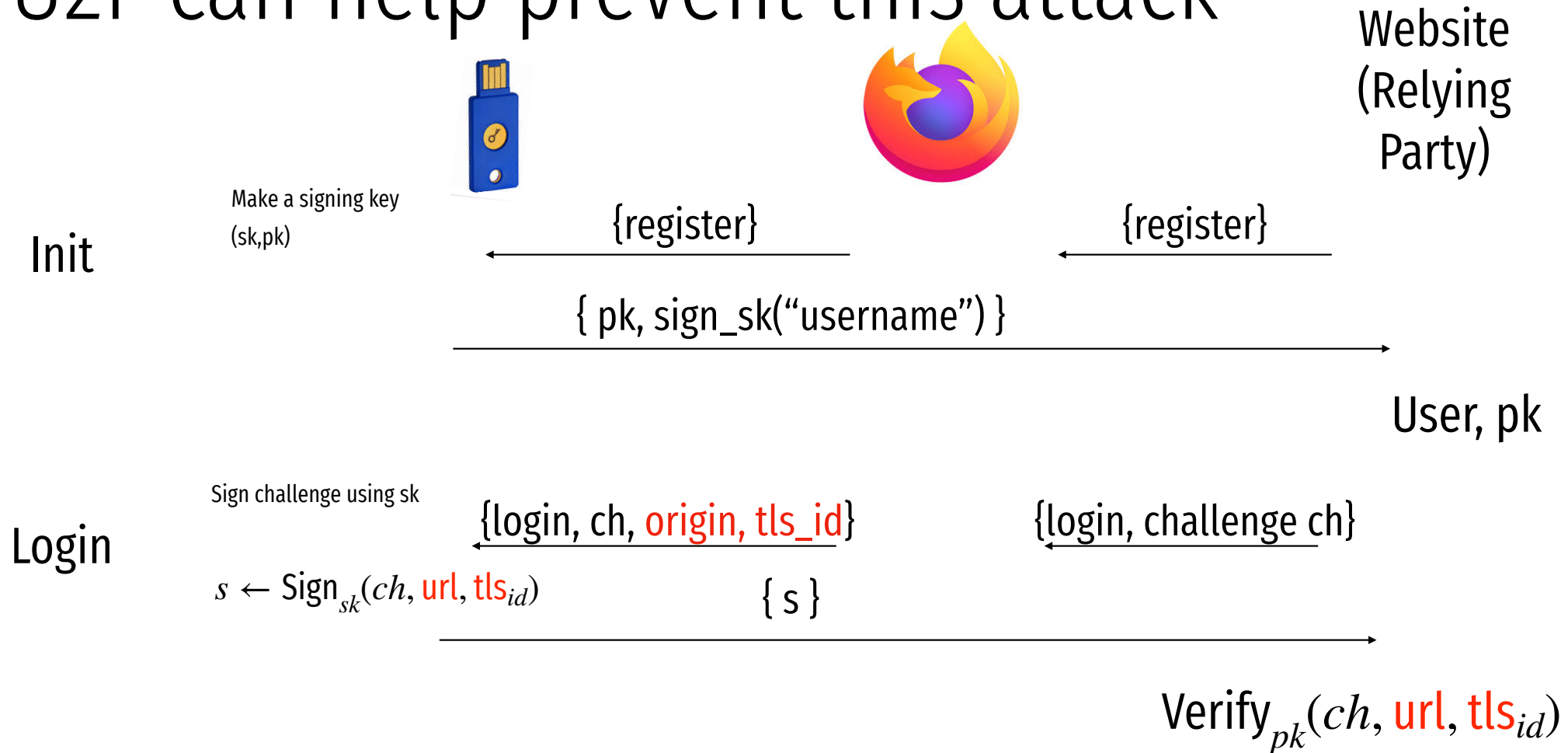
[MY ACCOUNT](#)

http://myaccount.google.com-securitysettingpage.tk/security/signinoptions/password?
e=am9obi5wb2Rlc3RhQGdtYWlsLmNvbQ%3D%3D&fn=Sm9obiBQb2Rlc3Rh&n=Sm9obg%3
D%3D&img=Ly9saDQuZ29vZ2xldXNlcmNvbniRlbnQuY29tLy1RZVIPbHJkVGp2WS9BQUFB...

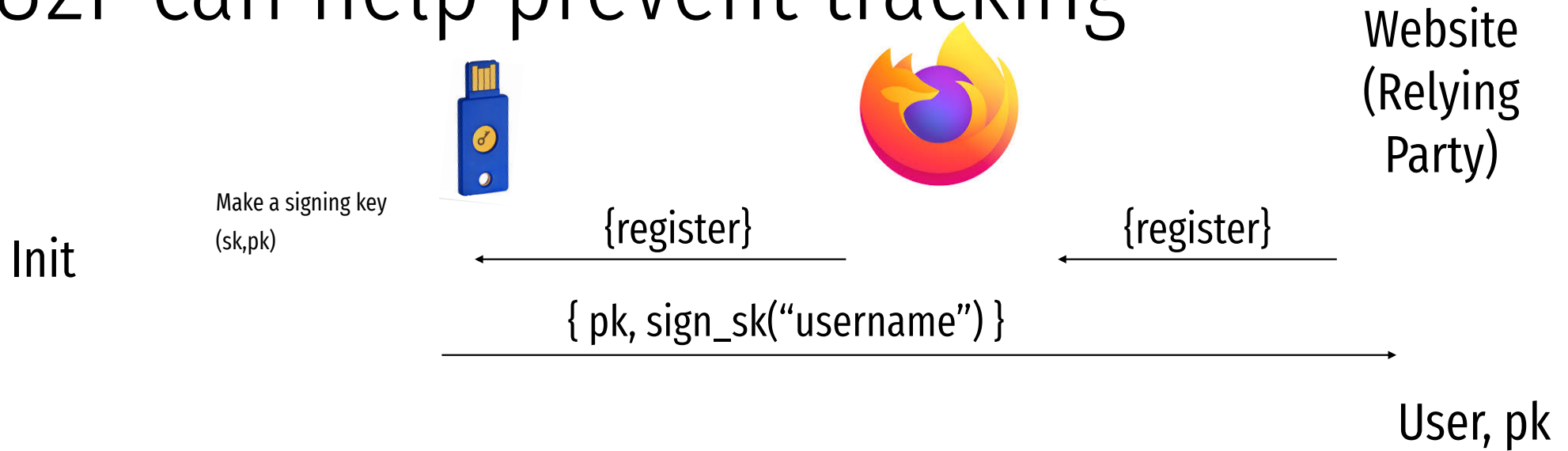
e=am9obi5wb2Rlc3RhQGdtYWlsLmNvbQ%3D%3D&f=Sm9obiBQb2Rlc3Rh&n=Sm9obg%3D%3D&img=Ly9saDQuZ29vZ2xldXNlcmNvb3RlbnQuY29tLy1RZVlPbHJkV2p2WS9BQUFBQUFBQUFBSS9BQUFBQUFBQUFBCTs9CQldVOVQ0bUZlUWS9waG90by5qcGc%3D&id=1sutlodlwe

bitly.com/ [redacted] COPY

U2F can help prevent this attack



U2F can help prevent tracking



U2F can help prevent tracking

Init

Make a signing key
(sk,pk)
And link it with
appid, and create
A token "h"



Website
(Relying
Party)

{appid, register}

{appid, register}

{ h, pk, sign_sk("username") }

User, h, pk

U2F can help prevent tracking



Website
(Relying
Party)

Init

Make a signing key
(sk,pk)
And link it with
appid, and create
A token "h"

$\{\text{appid, register}\}$

$\{\text{appid, register}\}$

$\{h, pk, \text{sign_sk}(\text{"username"})\}$

User, h, pk

Login

Lookup sk using h
Sign challenge using sk

$\{\text{login, h, ch, origin, tls_id}\}$

$\{\text{login, appid, challenge ch}\}$

$s \leftarrow \text{Sign}_{sk}(ch, \text{url}, \text{tls_id})$

$\{s, h\}$

$\text{Verify}_{pk}(ch, \text{url}, \text{tls_id})$
Check h

Sending request with appId: https://u2f.bin.coffee

```
{
  "version": "U2F_V2",
  "challenge": "uQnl3M4Rj3FZgs6WjyLaZAfwRh4"
}
```

Got response:

```
{
  "clientData": "eyJjaGFsbGVuZ2UiOiJlUW5sM000UmozRlpnczZXanlMYVpBZndSaDQiLCJvcmlnaW4iOiJodHRwciovL3UyZi5iaW4uY29mZmVlIiwidHlwIjoibmF2",
  "errorCode": 0,
  "registrationData": "BQRSuRLPv0p5udQ55vVhucf3N50q6...",
  "version": "U2F_V2"
}
```

Key Handle: 0r0Z0p0F0E0-0d0W0c0Q0b0X0i020C0w0-0E0v0h0t0T0T0P0_0-090_0a050P0e030u0b0z0l0K0Q0r000f0u030_0P020B0J0M0x0D050J0_0d0P0Q0e0j0

Certificate: 3082021c3082...

Attestation Cert

Subject: Yubico U2F EE Serial 14803321578

Issuer: Yubico U2F Root CA Serial 457200631

Validity (in millis): 1136332800000

Attestation Signature

R: 00b11e3efe5ae5ac7ca0e0d4fe2c5b5cf18a2531c0f4f70b11c30b72b5f946a9a3

S: 0f37ab2d4f93ebcdaed0a51b4b17fb93403db9873f0e9cce36f17b1502734bb2

[PASS] Signature buffer has no unnecessary bytes.: 71 == 71

[PASS] navigator.id.finishEnrollment == navigator.id.finishEnrollment

[PASS] uQnl3M4Rj3FZgs6WjyLaZAfwRh4 == uQnl3M4Rj3FZgs6WjyLaZAfwRh4

[PASS] https://u2f.bin.coffee == https://u2f.bin.coffee

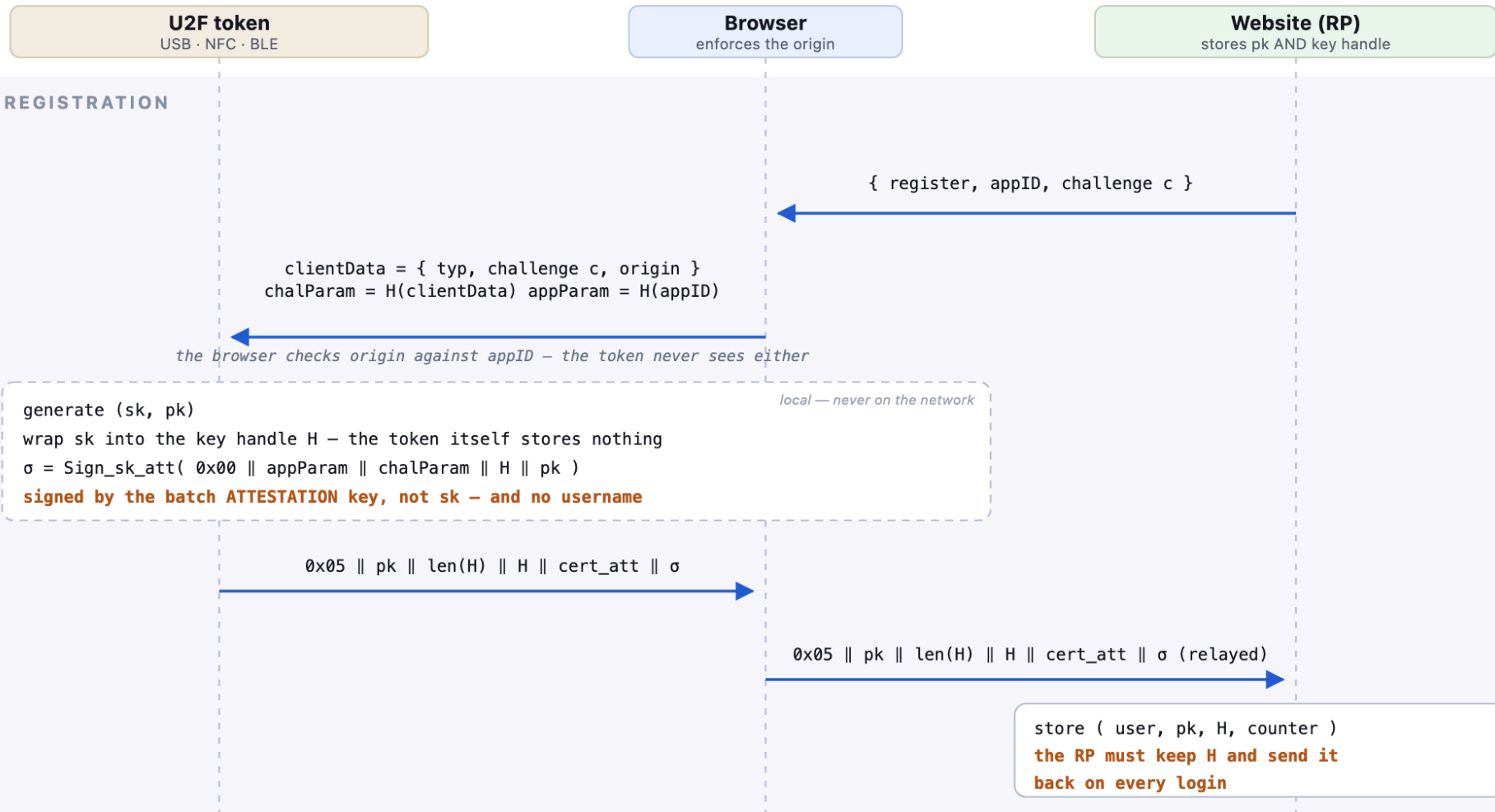
[PASS] Verified certificate attestation signature

[PASS] Imported credential public key

Failures: 0 TODOs: 0

FIDO U2F (2014) · CTAP1 — a second factor only

The token is stateless: it hands the RP a **key handle** and the RP must hand it back on every login.



AUTHENTICATION

{ challenge c, keyHandle H }

the RP must already know who you are – password first

clientData = { typ, challenge c, origin }
chalParam = H(clientData) , appParam , H

the browser re-checks origin against appID

unwrap sk from H; reject if appParam does not match
user presence: a touch – no biometric, no PIN
counter++
 $\sigma = \text{Sign_sk}(\text{appParam} \parallel \text{UP} \parallel \text{counter} \parallel \text{chalParam})$

UP || counter || σ

UP || counter || σ (relayed)

Verify_pk(appParam || UP || counter
|| chalParam)
check that the counter increased

Other problems

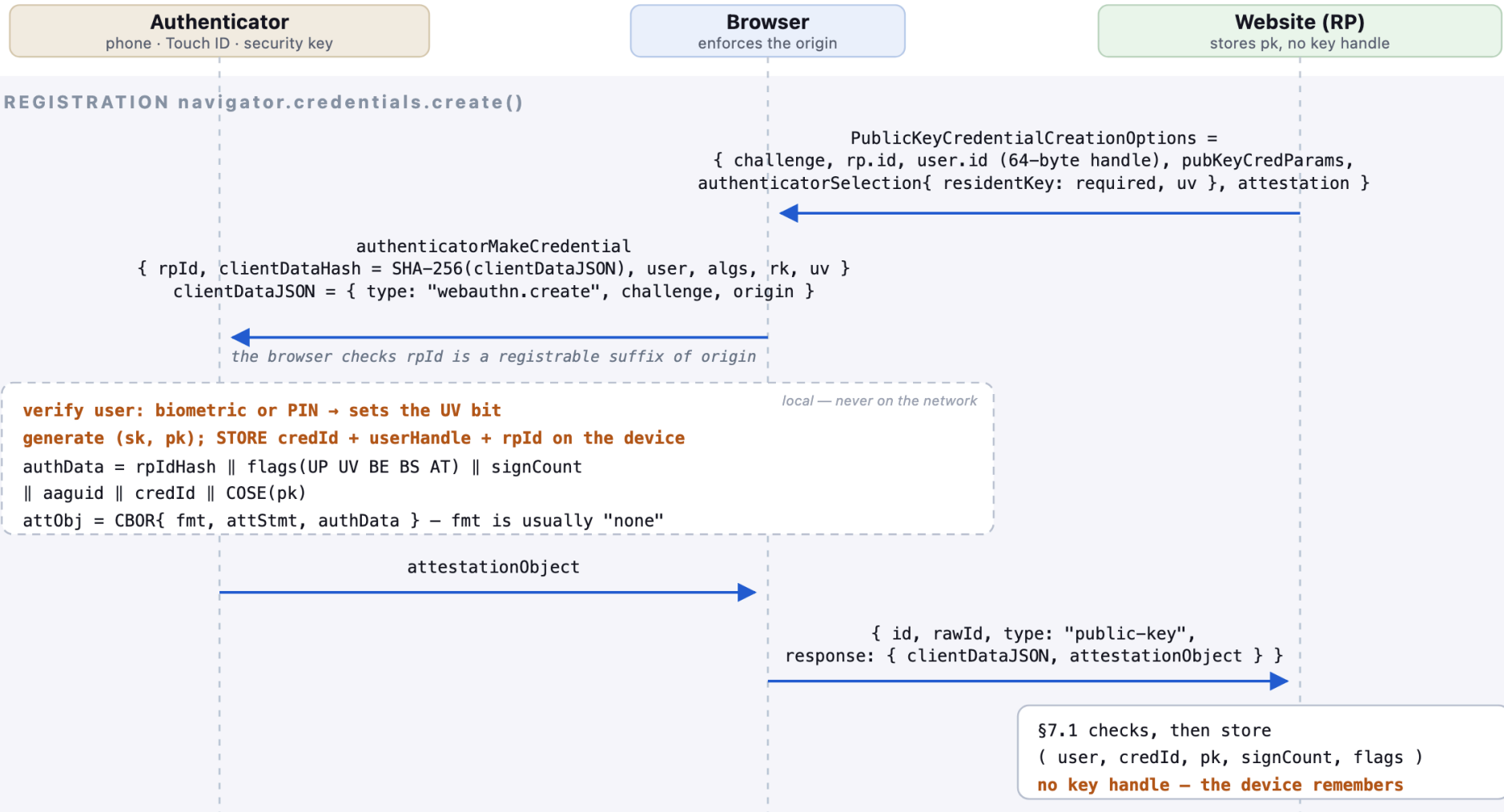
Need a username

Fixed crypto

Updated in “webauthn” standard

WebAuthn (W3C L2; L3 in progress) + CTAP 2.1/2.2 — “passkey”

Discoverable credential: the **authenticator** stores it, so no username is ever typed and no handle is returned.



AUTHENTICATION navigator.credentials.get()

PublicKeyCredentialRequestOptions =
{ challenge, rpId, userVerification }

allowCredentials is EMPTY – the RP does not know who is logging in

authenticatorGetAssertion
{ rpId, clientDataHash, no allowList, uv }
clientDataJSON = { type: "webauthn.get", challenge, origin }

the same origin check, in the same place

list the resident credentials for rpId; the user picks one

verify user: biometric or PIN

signCount++ – stays 0 for synced passkeys, so the clone check is gone

σ = Sign_sk(authData || SHA-256(clientDataJSON))

authData , σ , userHandle

{ rawId, response: { clientDataJSON, authenticatorData,
signature, userHandle } }

find the account by userHandle

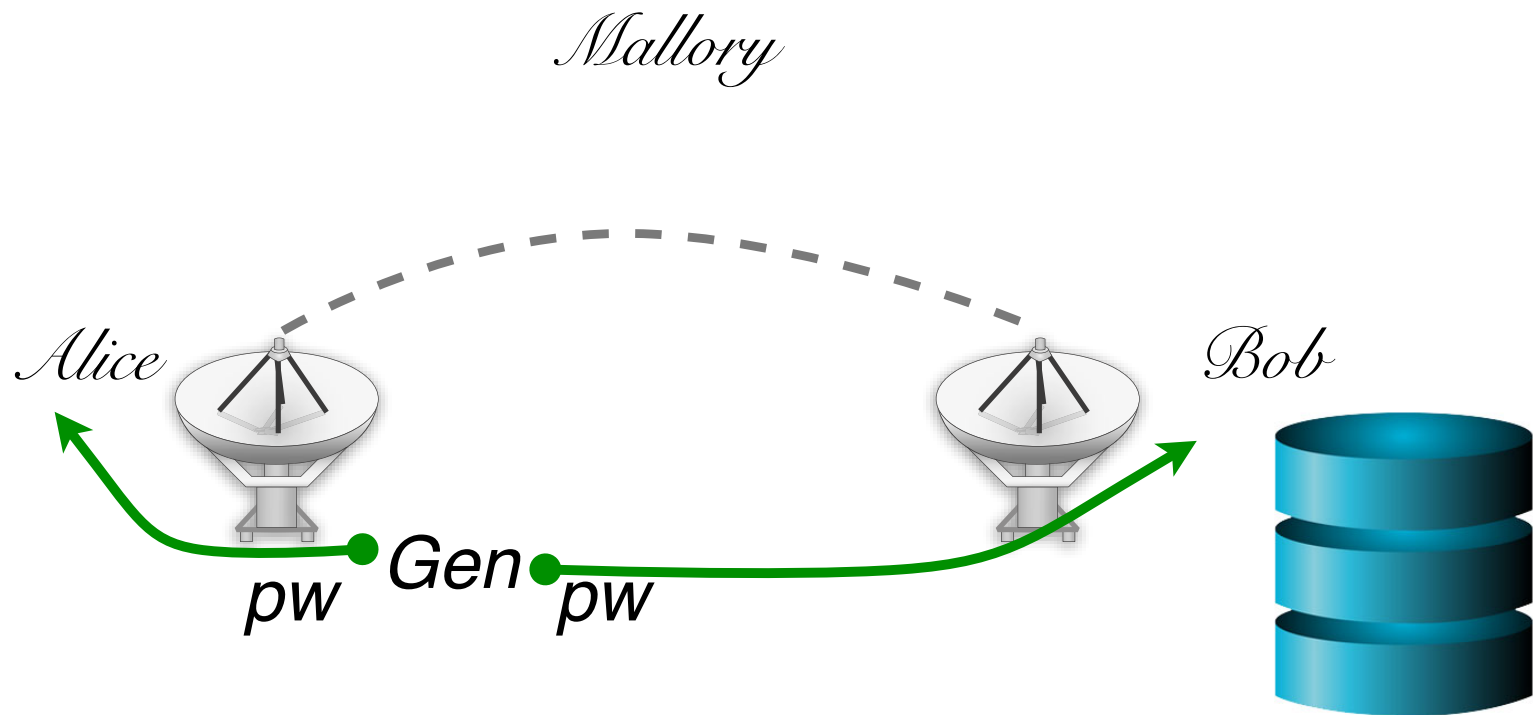
Verify_pk(authData || H(clientDataJSON))

\$7.2 checks: origin, challenge, flags

Same defence, same shape. What actually changed: UV makes the passkey a FIRST factor, so there is no password; discoverable credentials remove the username; attestation is optional and usually "none"; crypto is negotiated rather than fixed at P-256; and the BE/BS flags admit the key may be synced — moving the weak link to recovery.

Future without passwords?

Password Security game

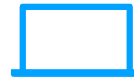


More realistic picture of the world

Alice
pw



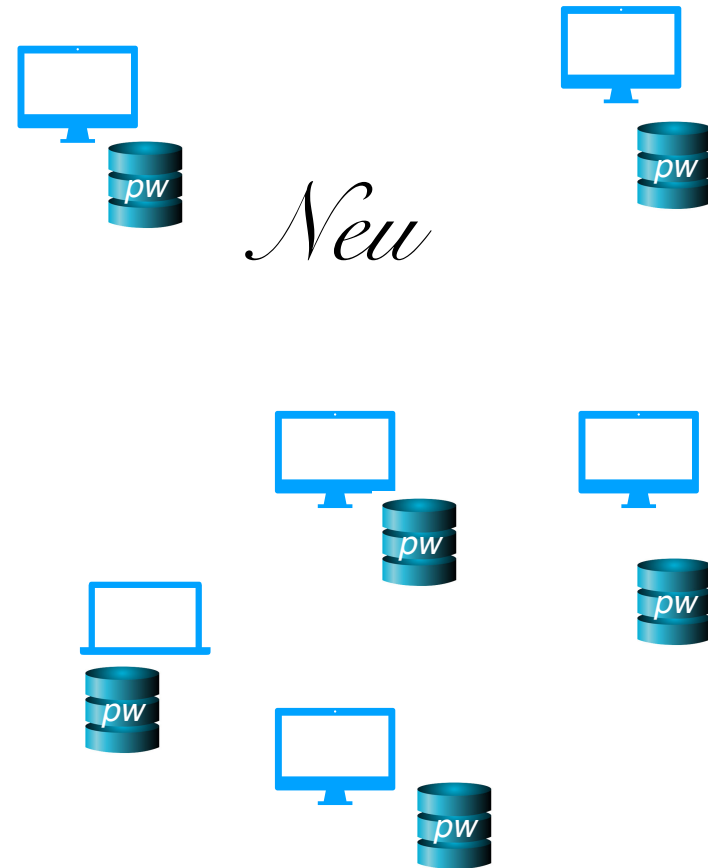
New



More realistic picture of the world

What are the problems with this solution?

Alice
pw



The problem of distributed authentication

Alice
pw

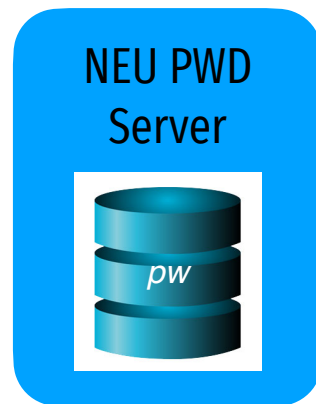


Distributed authentication: Attacker model

What can attacker do?



Alice
pw



Distributed authentication: Bad Solution

What can attacker do?

