
OWASP GLOBAL AppSec
VIE '26
NNA JUN
25-26
25
years
of open source security

OWASP GLOBAL AppSec

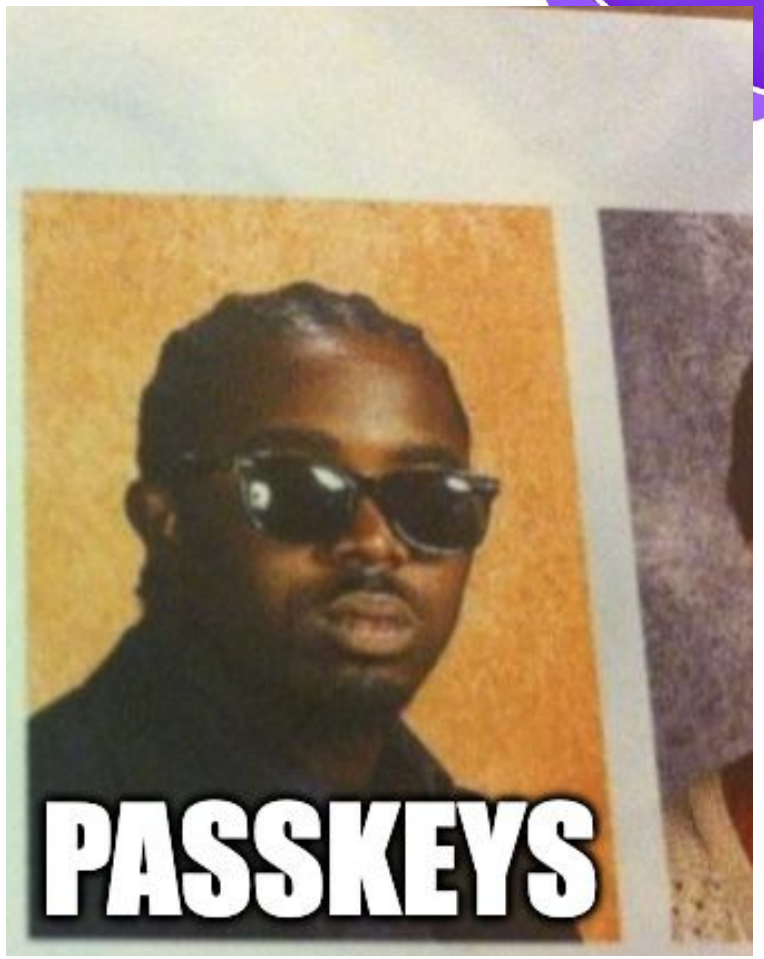
VIENNA '26 JUN 25-26

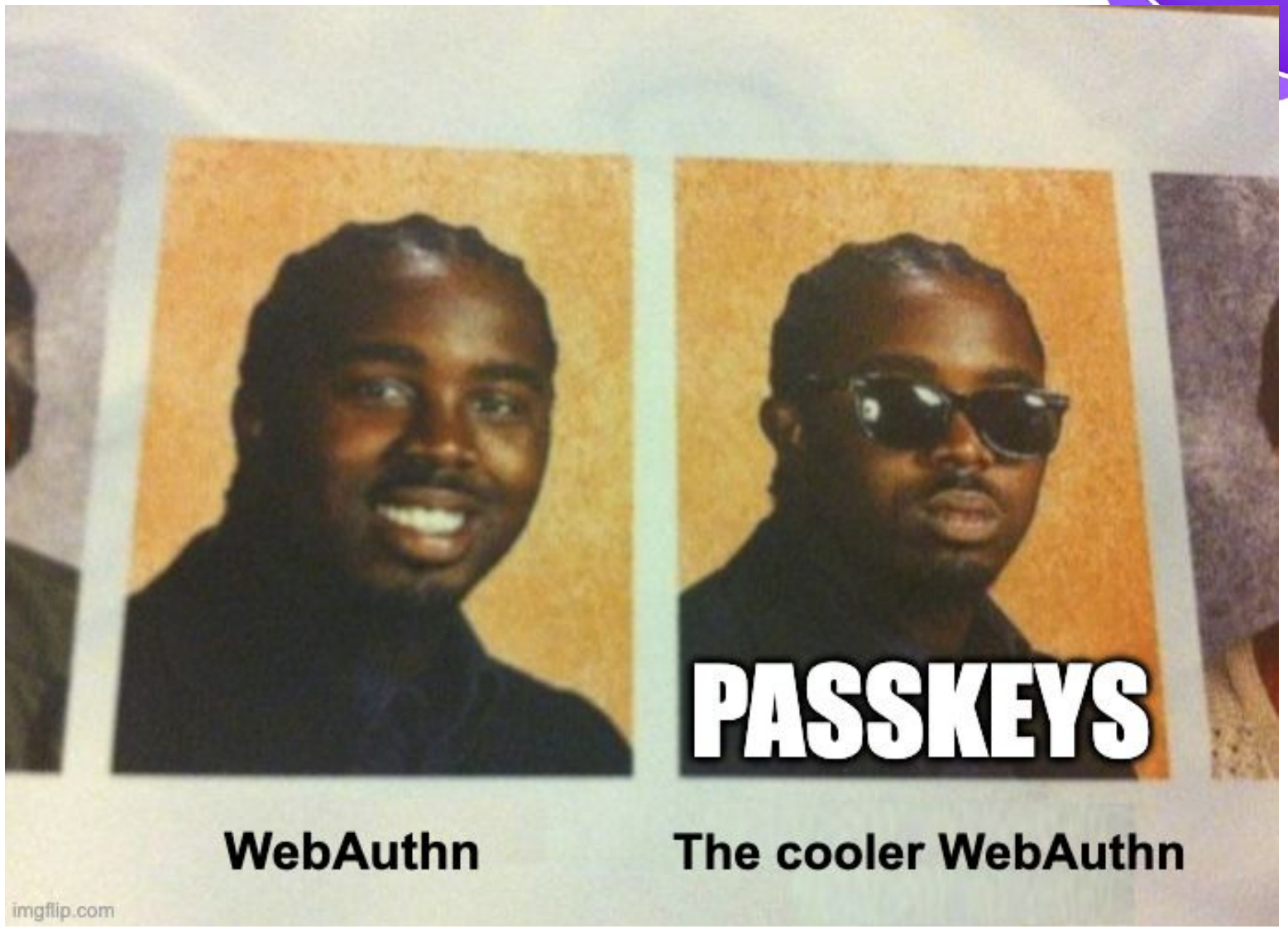
25 years of open source security

Phishing for Passkeys - An Analysis of WebAuthn and CTAP

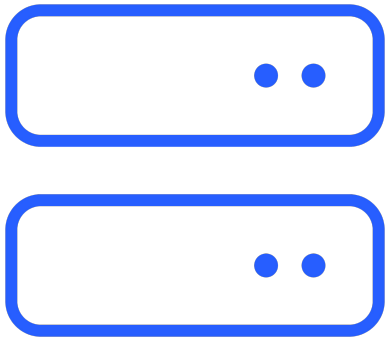
Michael Kuckuk

PHISHING FOR PASSKEYS




PASSKEYS
WebAuthn
The cooler WebAuthn
imgflip.com

Relying Party (RP)



- Creates crypto challenge etc.
- Stores public keys
- Verifies login
- 🤨 *Declares RP ID*

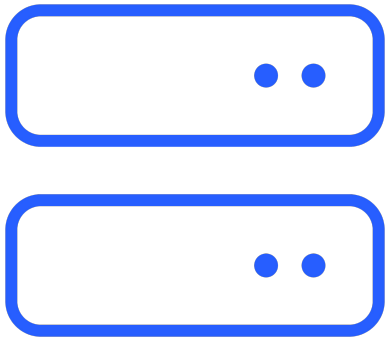
Client

- "Forwards" messages ↔
- 🤨 *Gatekeeper*

Authenticator

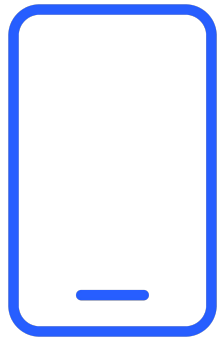
- Creates & stores keys
- Signs challenge
- Performs 2nd factor

Relying Party (RP)



Client

Authenticator



Relying Party (RP)

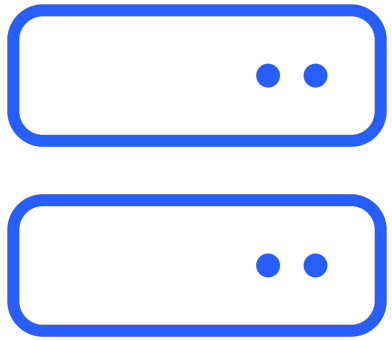
Client

Authenticator

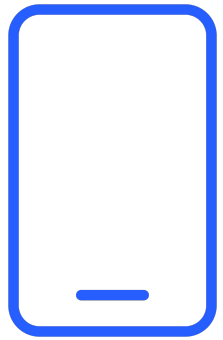
Relying Party (RP)

Client

Authenticator



=

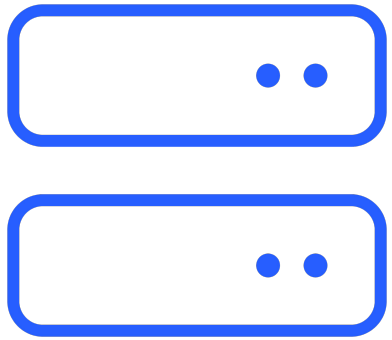


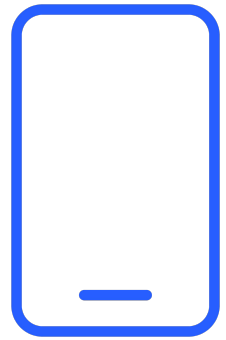
"Platform-attached Authenticator"

Relying Party (RP)

Client

Authenticator

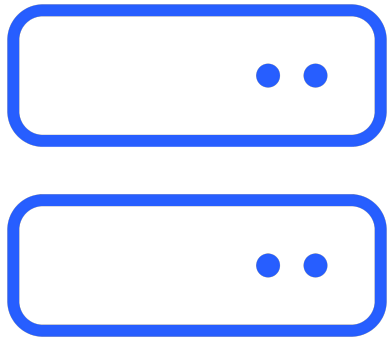




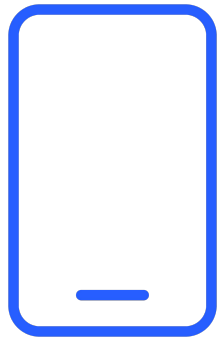
Relying Party (RP)

Client

Authenticator



≠

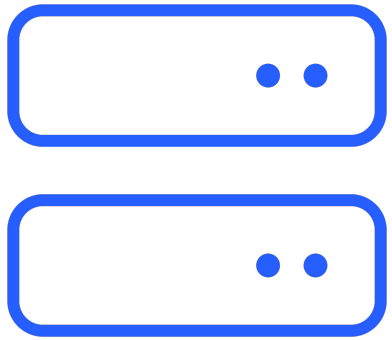


"Cross-platform-attached Authenticator"

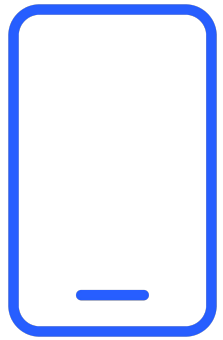
Relying Party (RP)

Client

Authenticator



≠

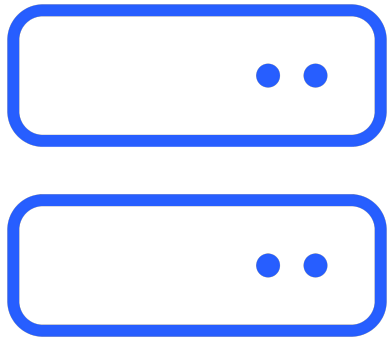


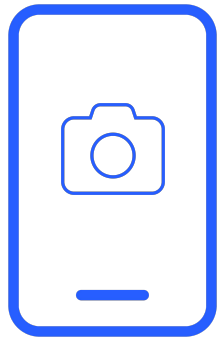
Cross-Device Authentication (CDA)

Relying Party (RP)

Client

Authenticator



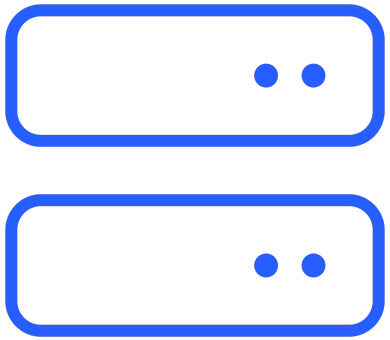


"Cross-platform-attached Authenticator"

Relying Party (RP)

Client

Authenticator



"Cross-platform-attached Authenticator"

PHISHING FOR PASSKEYS

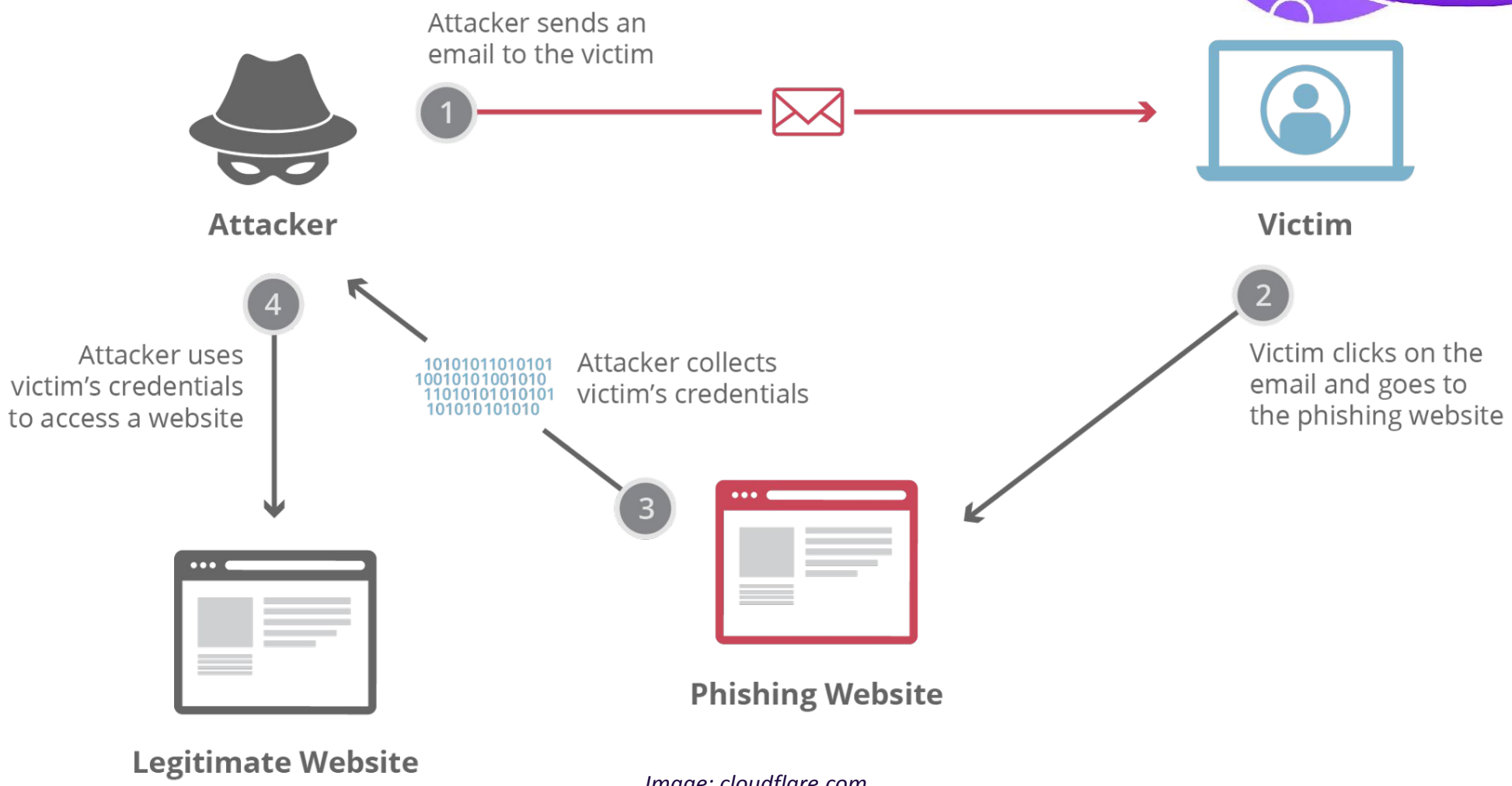


Image: cloudflare.com


Regular Page
Phishing Page
Corporate needs you to find the differences between this picture and this picture.
Humans
They're the same picture.
imgflip.com

PHISHING FOR PASSKEYS

Have I Been Pwned: Project operator Troy Hunt pwned

The operator of Have I Been Pwned was himself the victim of a phishing attack. The emails from the newsletter mailing list were stolen.

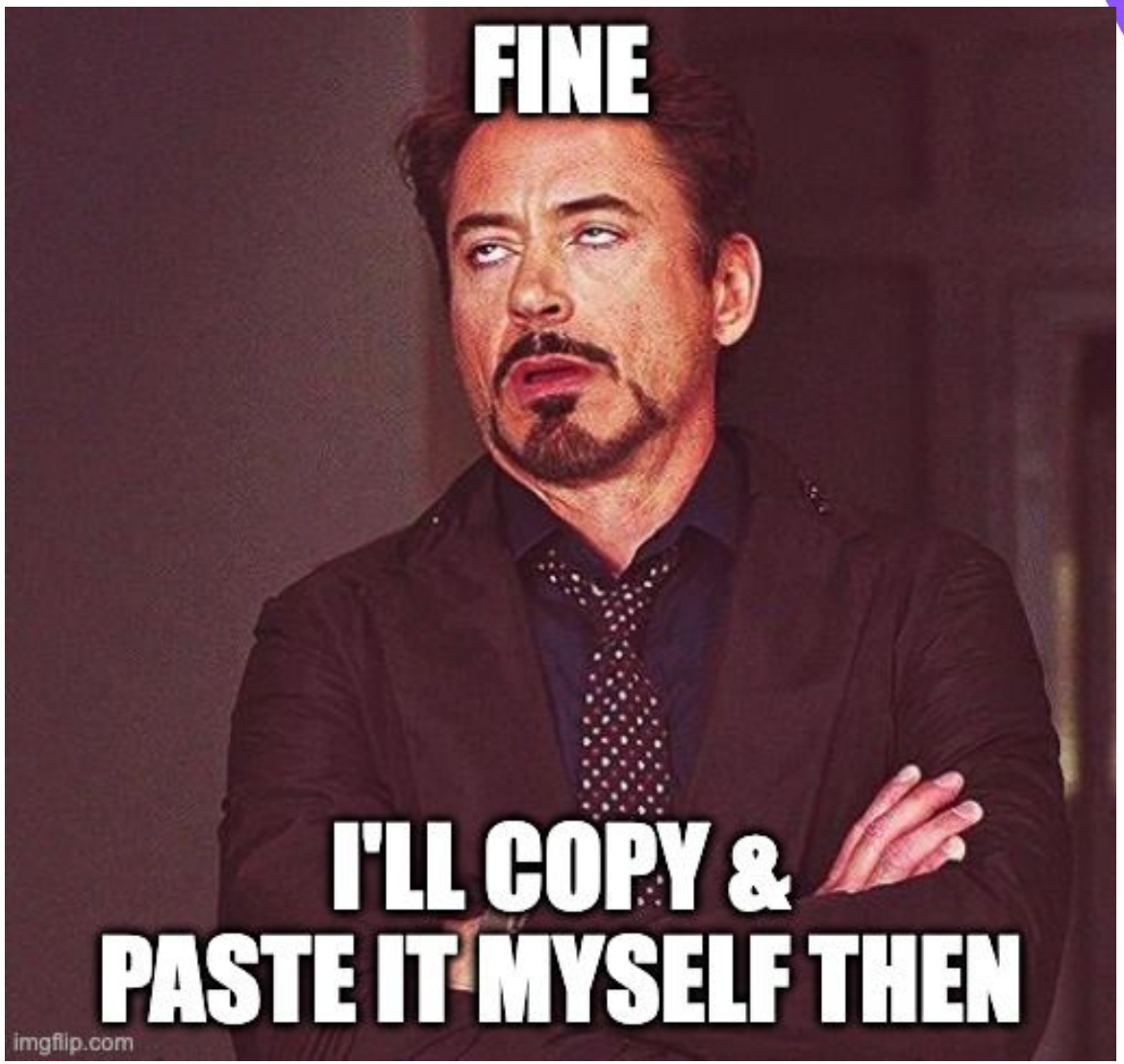
Have I Been Pwned? Yes! (Image: heise online / dmk)

Mar 26, 2025 at 9:18 pm CET 3 min. read | Security

By Dirk Knop


NOWASP.ORG
OWASP.ORG
PASSWORD MANAGERS
imgflip.com

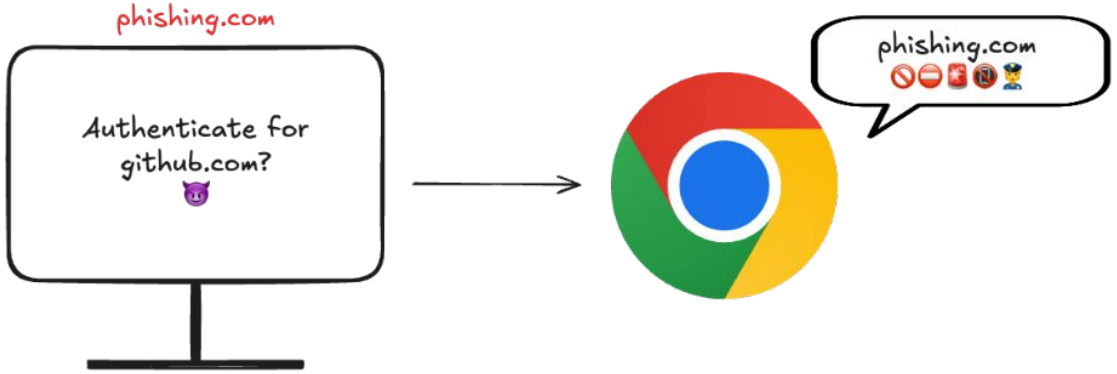
PHISHING FOR PASSKEYS



Resistance against phishing

- Server declares its Relying Party ID (RPID, usually the website's domain or superdomain)
- Browser checks if current origin matches that RPID
- Authenticator ignores credentials that belong to other RPIDs
- ‼️ RP verifies origin (provided by client, signed by authenticator)

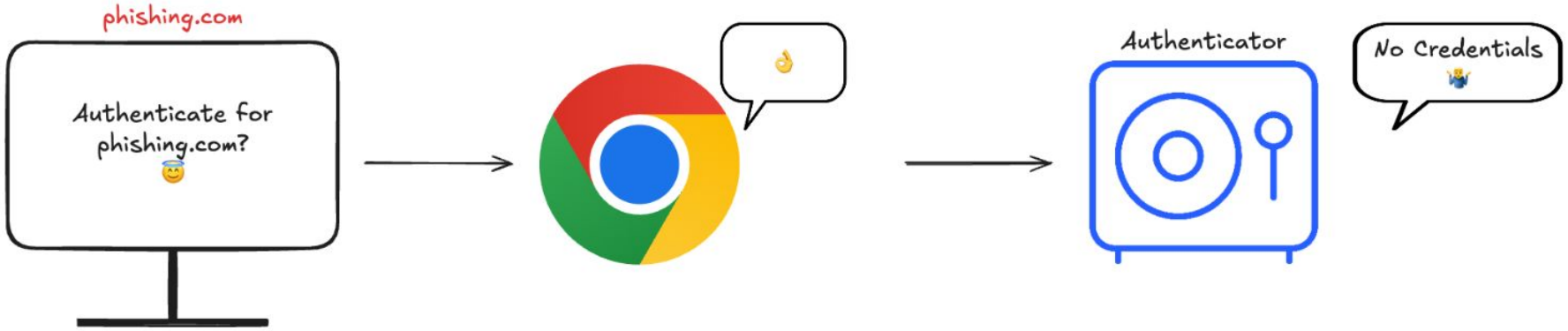
→ Users cannot use credentials on unintended (= phishing) websites!



Resistance against phishing

- Server declares its Relying Party ID (RPID, usually the website's domain or superdomain)
- Browser checks if current origin matches that RPID
- Authenticator ignores credentials that belong to other RPIDs
- ‼️ RP verifies origin (provided by client, signed by authenticator)

→ Users cannot use credentials on unintended (= phishing) websites!



Resistance against phishing

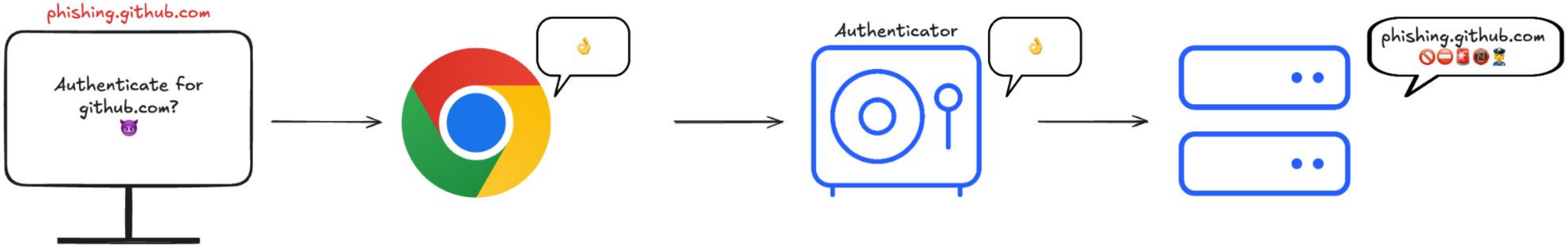
- Server declares its Relying Party ID (RPID, usually the website's domain or superdomain)
- Browser checks if current origin matches that RPID
- Authenticator ignores credentials that belong to other RPIDs
- ‼️ RP verifies origin (provided by client, signed by authenticator)

→ Users cannot use credentials on unintended (= phishing) websites!

Resistance against phishing

- Server declares its Relying Party ID (RPID, usually the website's domain or superdomain)
- Browser checks if current origin matches that RPID
- Authenticator ignores credentials that belong to other RPIDs
- ‼️ RP verifies origin (provided by client, signed by authenticator)

→ Users cannot use credentials on unintended (= phishing) websites!



PHISHING FOR PASSKEYS

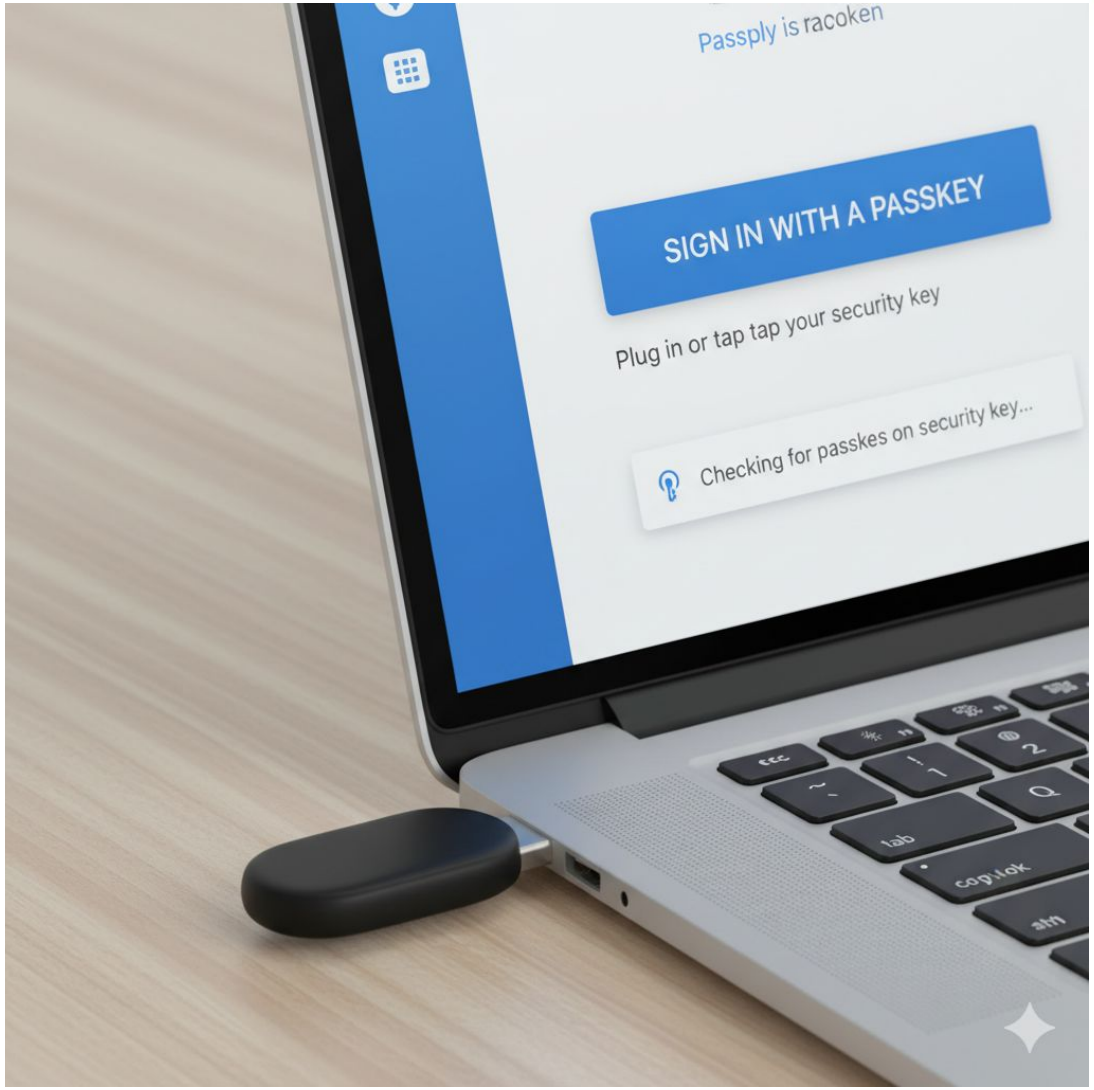
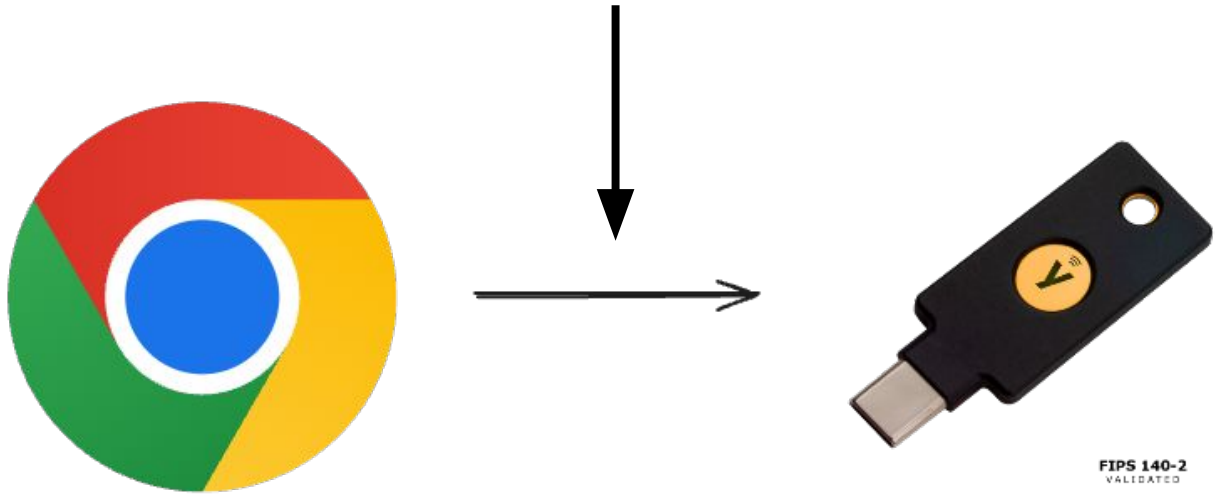


Image is AI generated

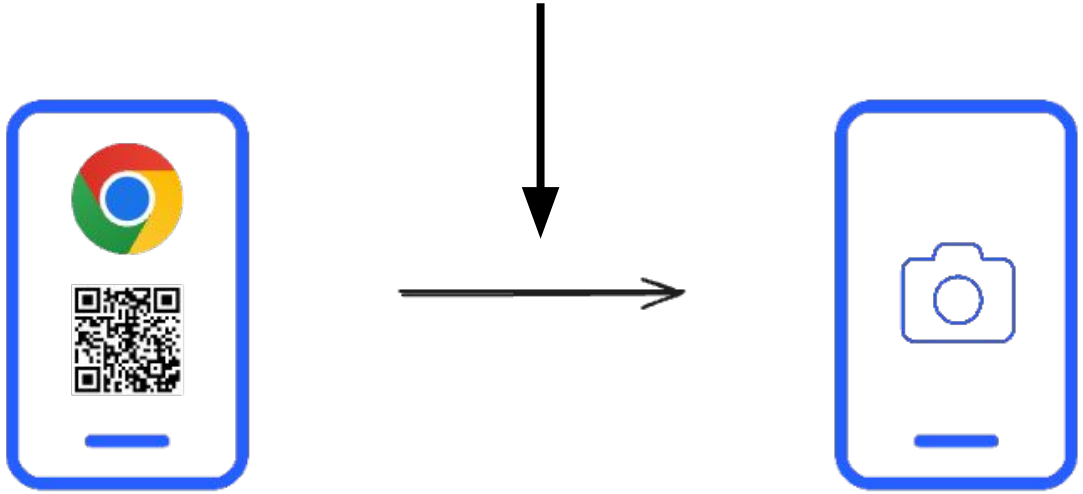
User can visually verify connection



PHISHING FOR PASSKEYS

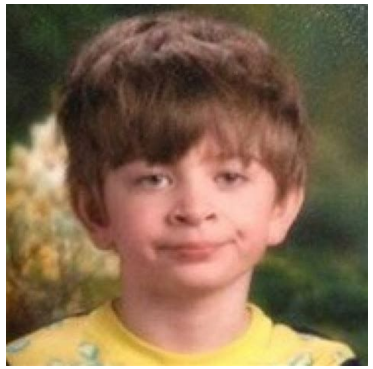
Image is AI generated

Is this magic?
Can we mess with it?

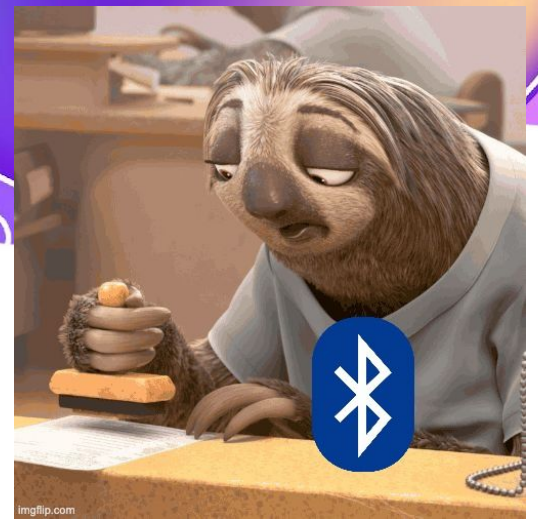


What WebAuthn specifies

- Client and Authenticator must exchange certain information
- Authenticator must be able to execute `user verification`
- The *how* is "intentionally unspecified" and "outside the scope of this specification"



PHISHING FOR PASSKEYS



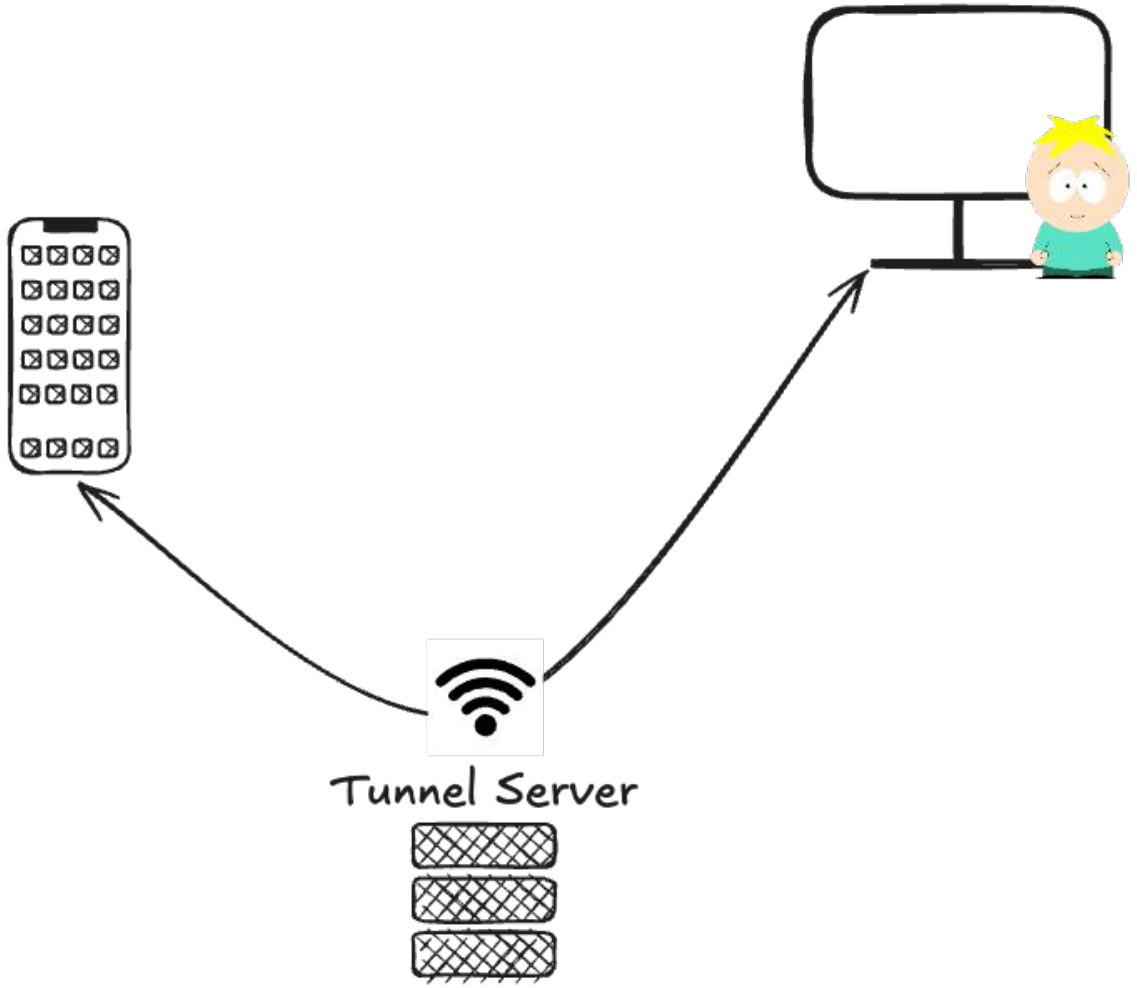
Client To Authenticator Protocol (CTAP)

- Only applies to Cross-Device Authentication
- Application level (what)
  - Many messages mirror WebAuthn (e.g. `authenticatorGetAssertion`)
  - Additional messages for implementing other requirements
- Transport level (how)
  - Specifies encoding
  - Ensures correct devices are communicating with each other
  - Transports*: USB, NFC, Bluetooth Low Energy (BLE)
    - ➢ Guarantees physical proximity

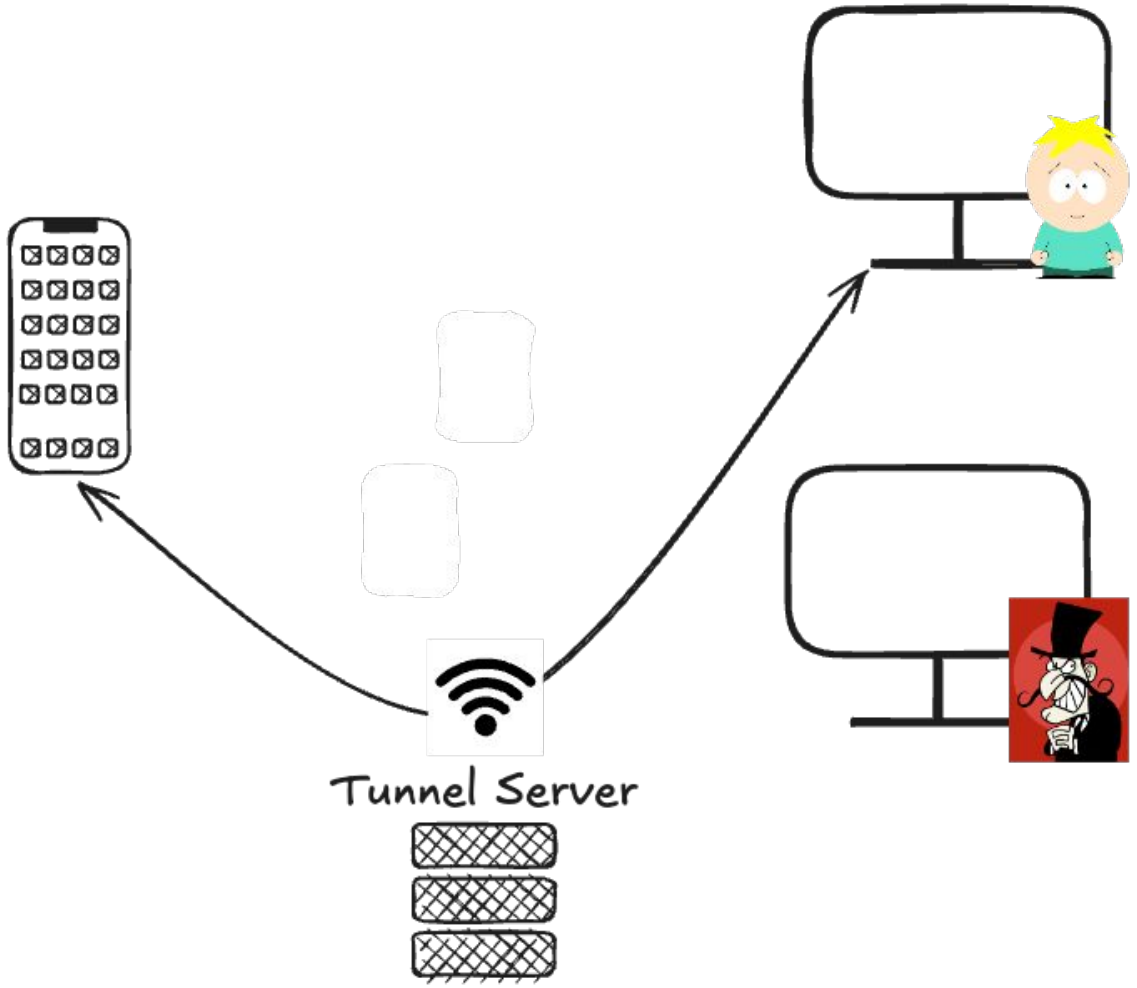
Image is AI generated

Re-inventing QR-initiated Hybrid Transport

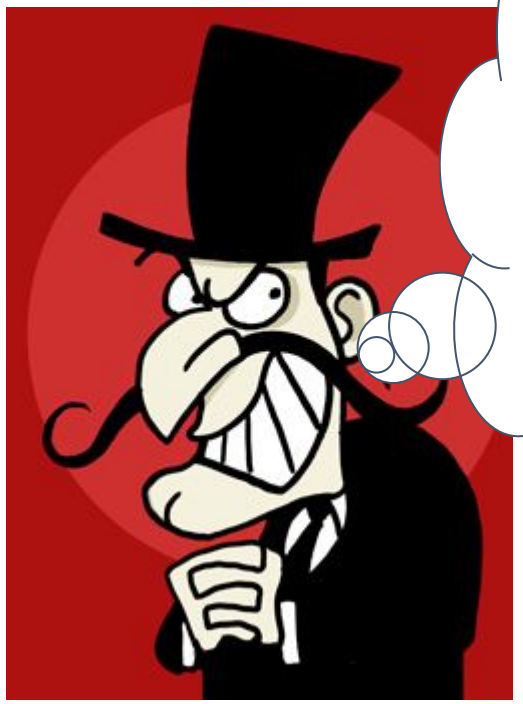
Re-inventing QR-initiated Hybrid Transport



Re-inventing QR-initiated Hybrid Transport



Re-inventing QR-initiated Hybrid Transport



I'll pretend to be the client through the Tunnel Server!

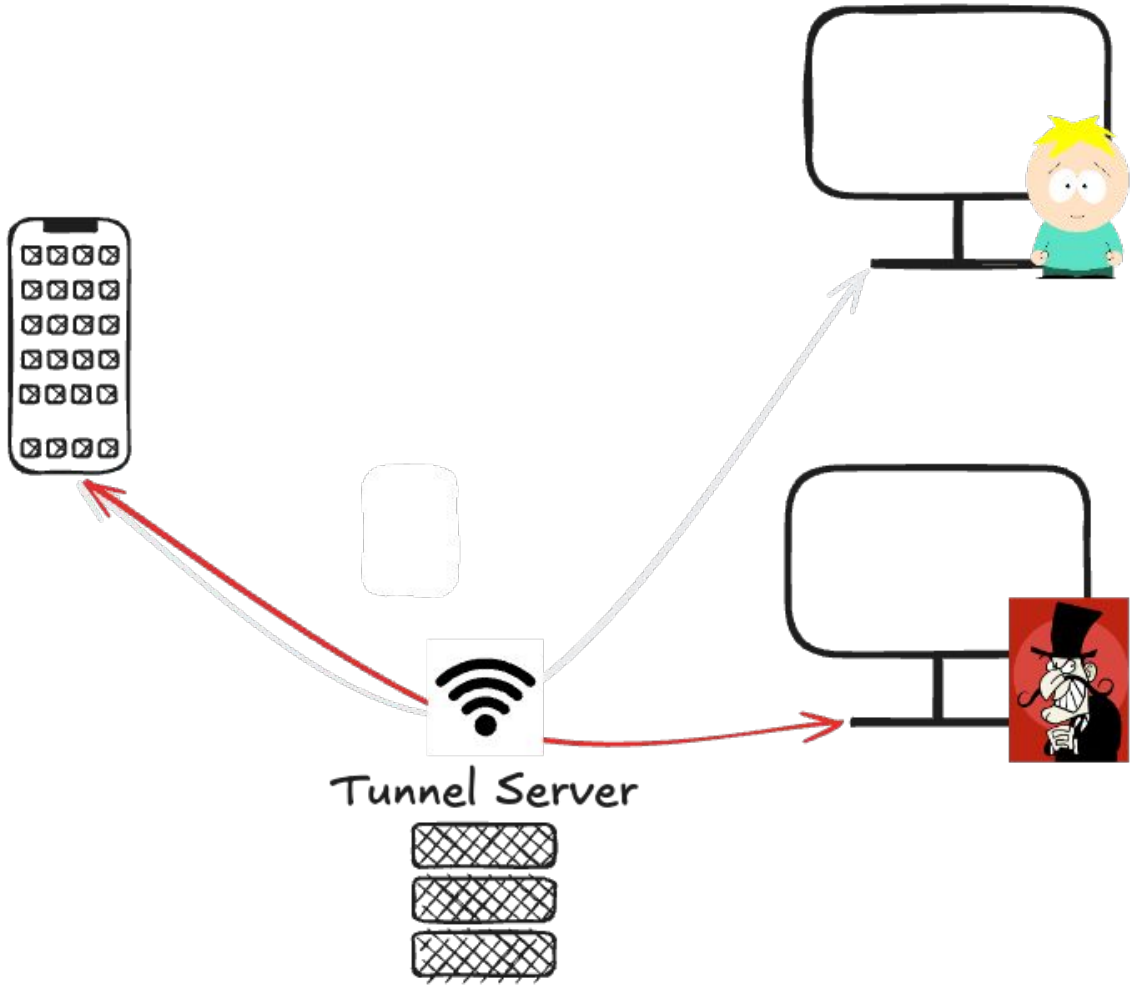
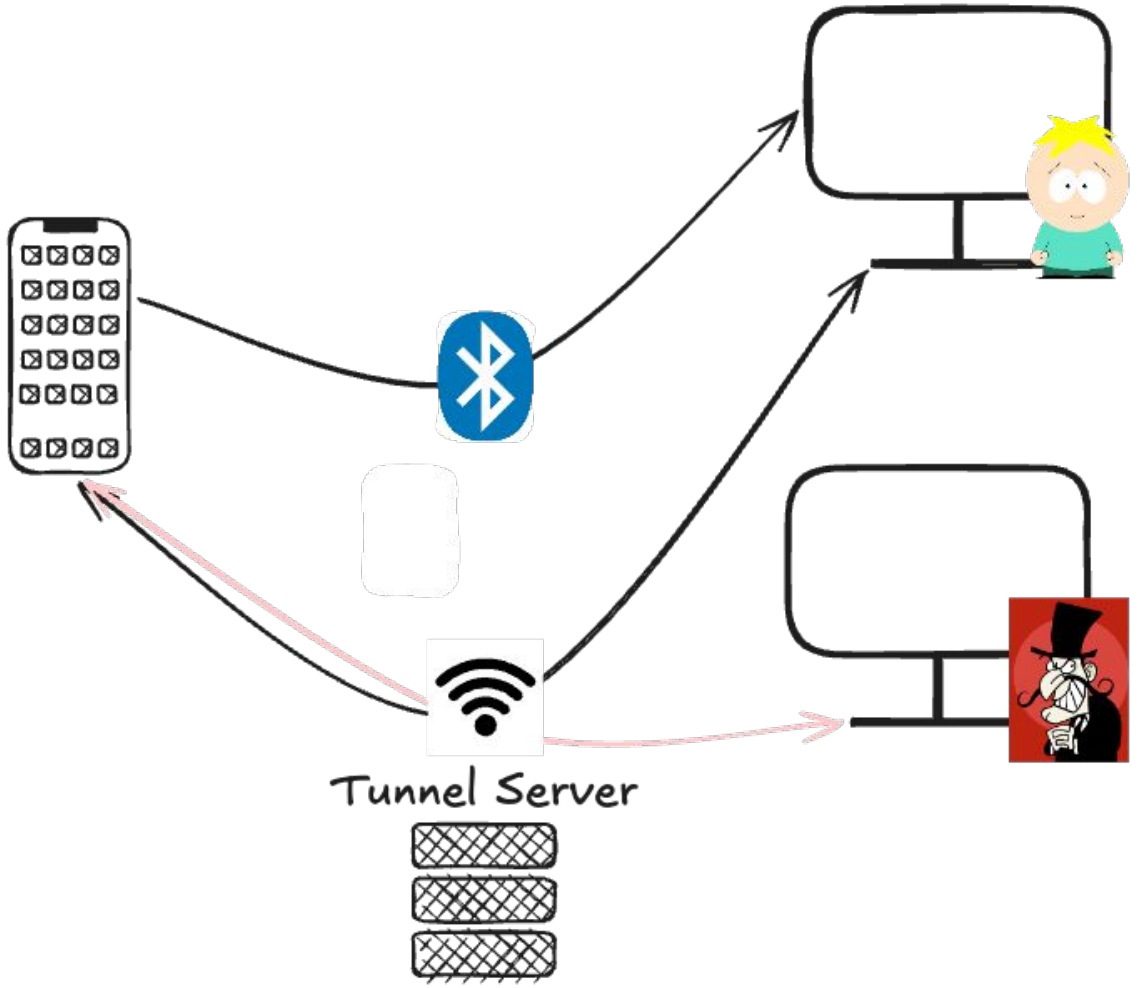
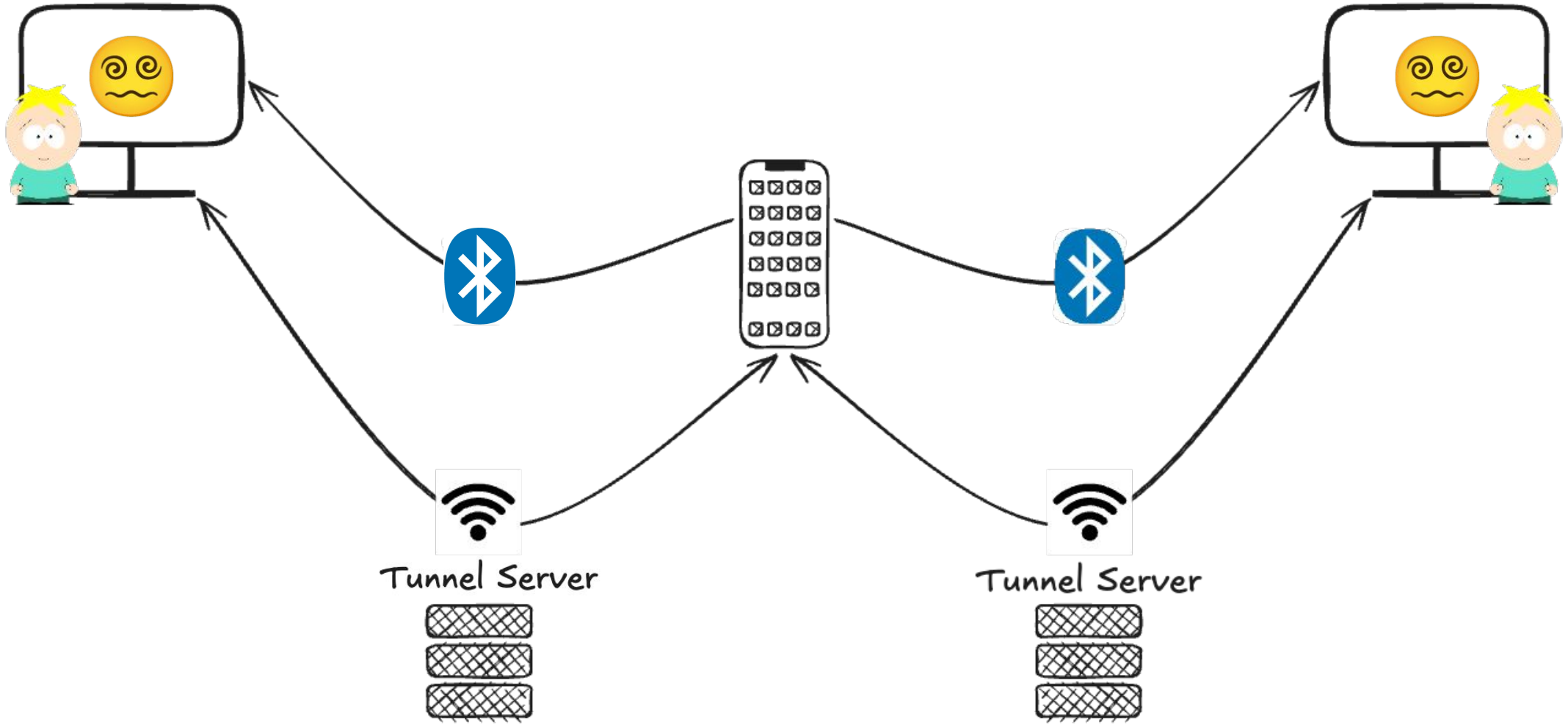


Image: J.J. at the English-language Wikipedia

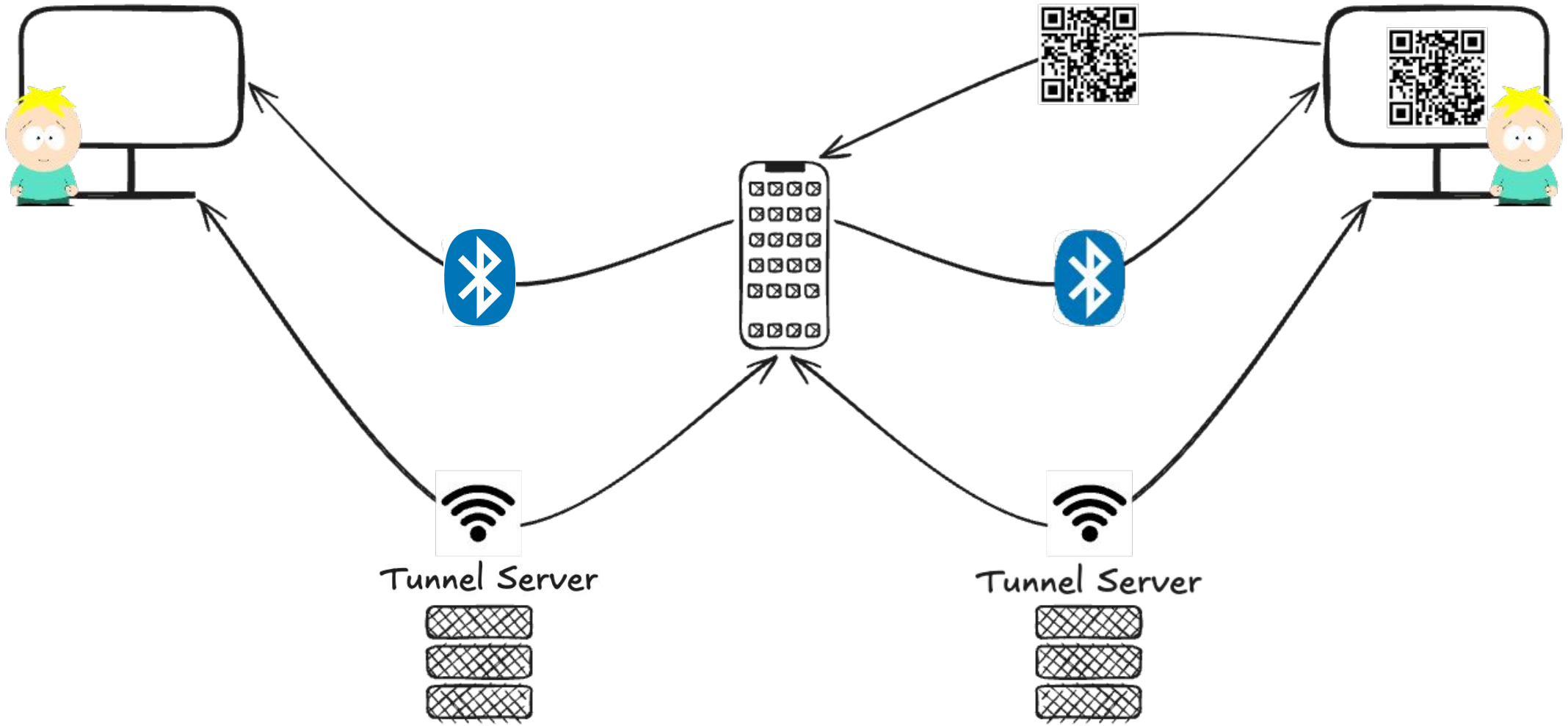
Re-inventing QR-initiated Hybrid Transport



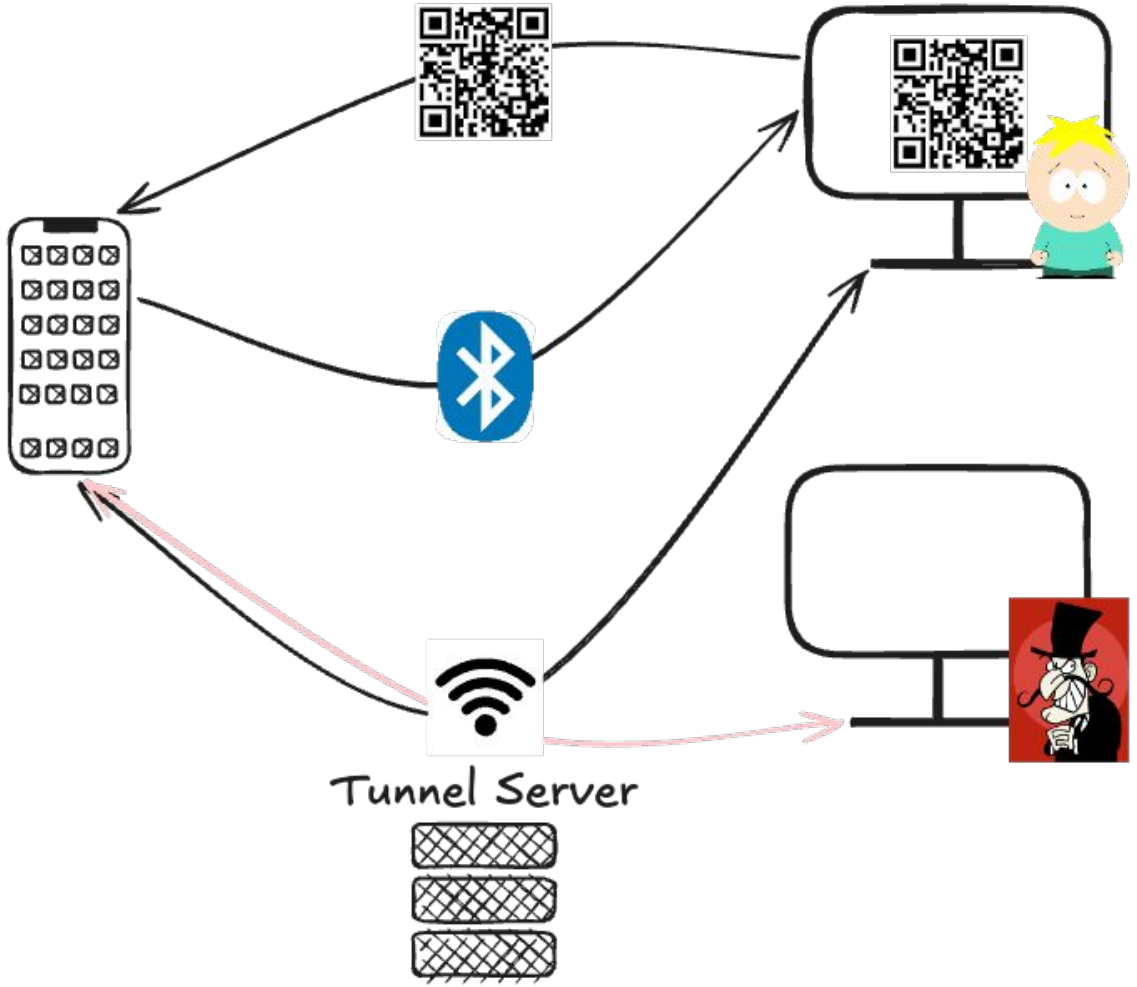
Re-inventing QR-initiated Hybrid Transport



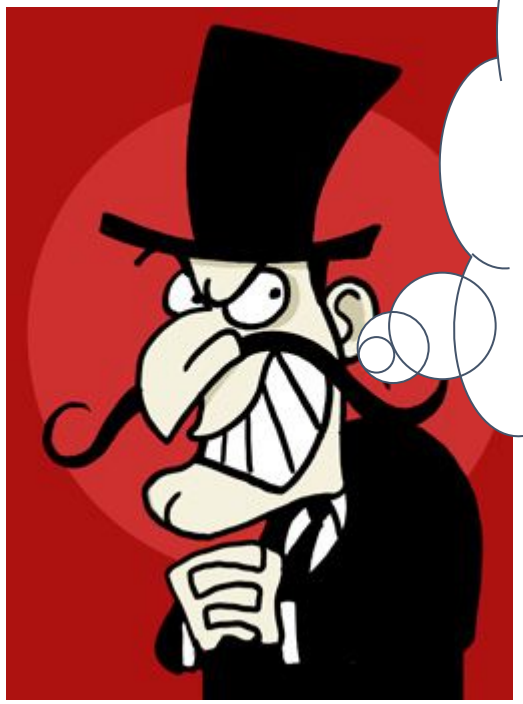
Re-inventing QR-initiated Hybrid Transport



Re-inventing QR-initiated Hybrid Transport



Re-inventing QR-initiated Hybrid Transport



I'll place a BLE beacon & listen to advert as well!

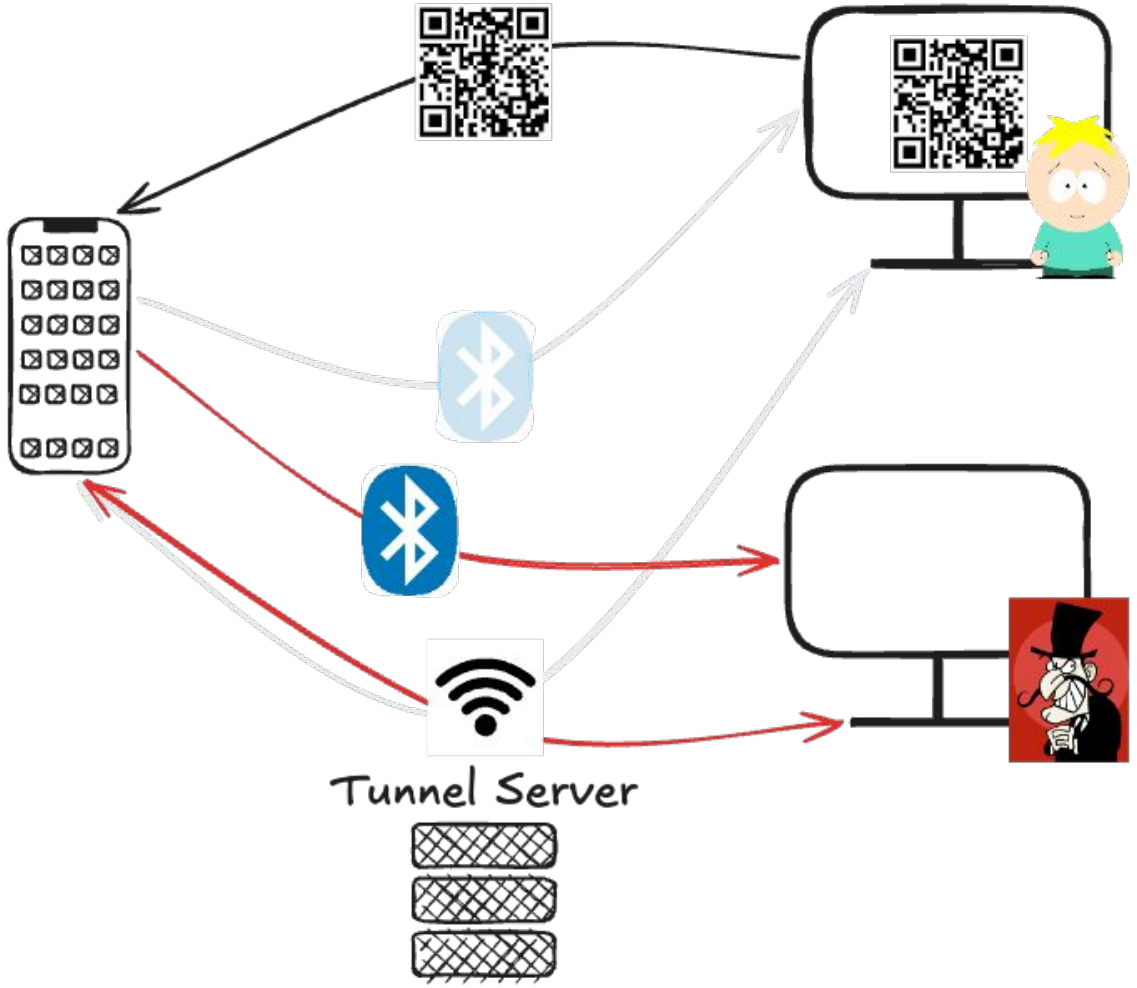
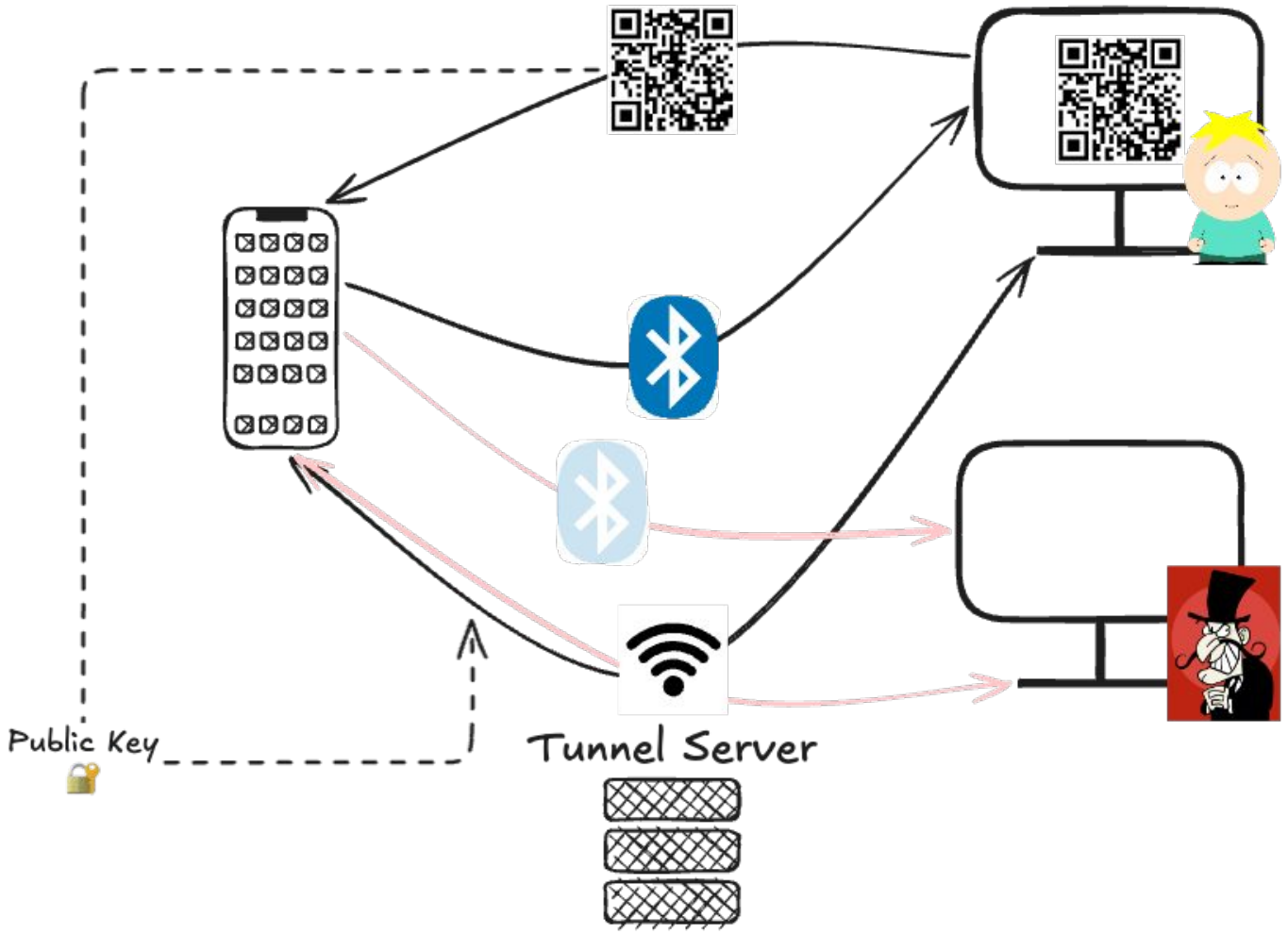


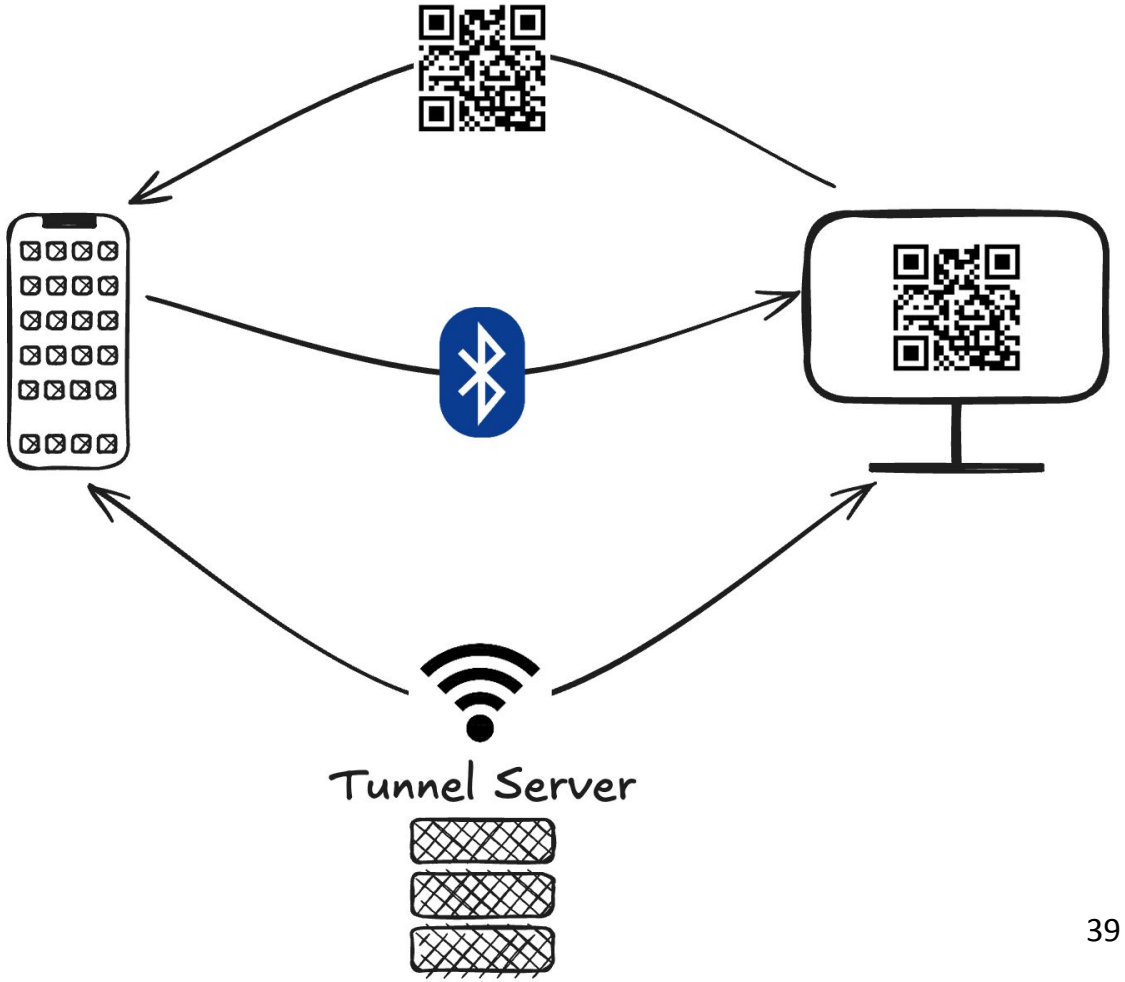
Image: J.J. at the English-language Wikipedia

Re-inventing QR-initiated Hybrid Transport

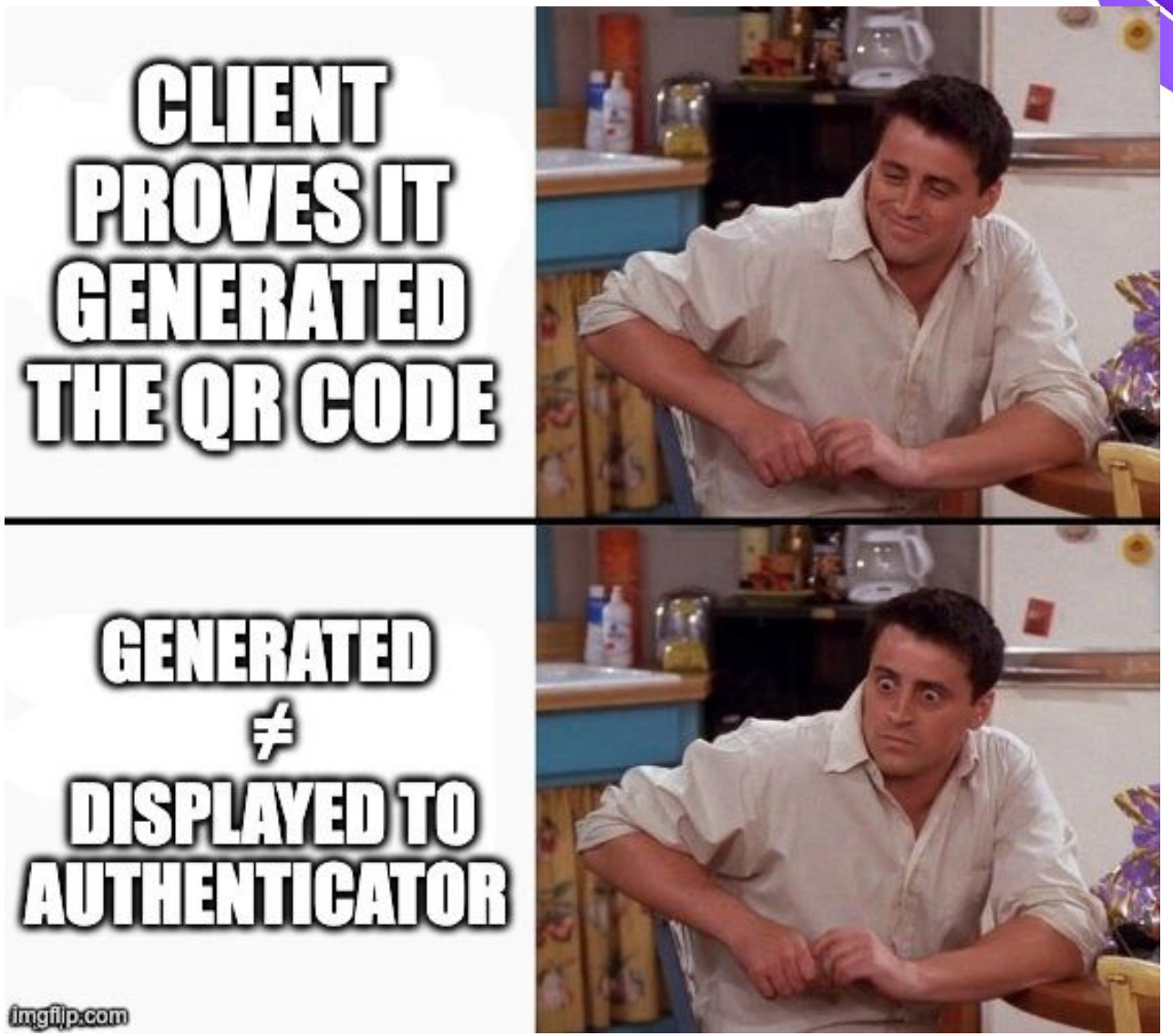


QR-initiated Hybrid Transport as per CTAP

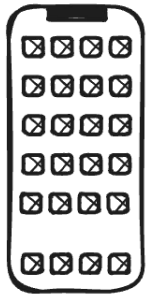
- … and much more
  - Tunnel Server negotiation
  - Encryption of tunnelled messages
  - Info for UI tweaks

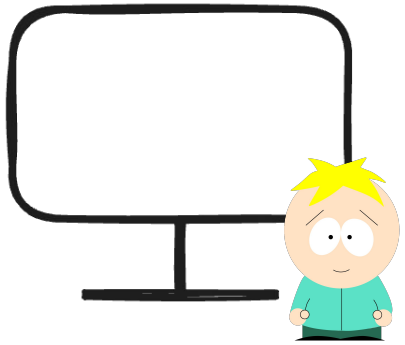


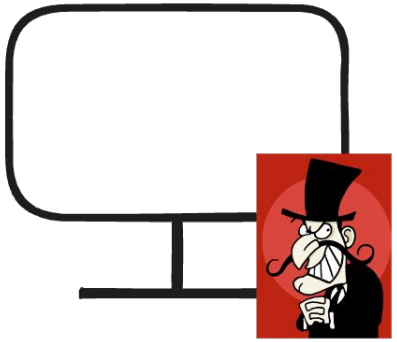

CLIENT
PROVES IT
GENERATED
THE QR CODE


CLIENT PROVES IT GENERATED THE QR CODE
GENERATED ≠ DISPLAYED TO AUTHENTICATOR
imgflip.com

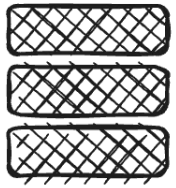
PHISHING FOR PASSKEYS



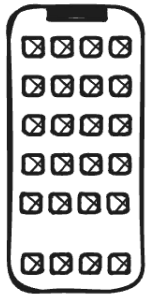


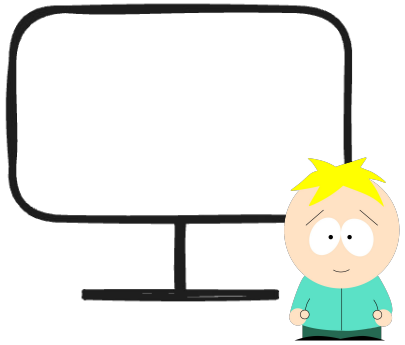


Tunnel Server

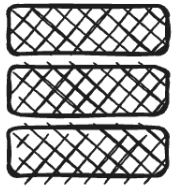


PHISHING FOR PASSKEYS

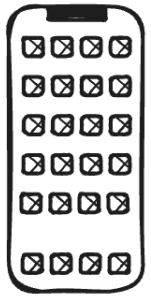


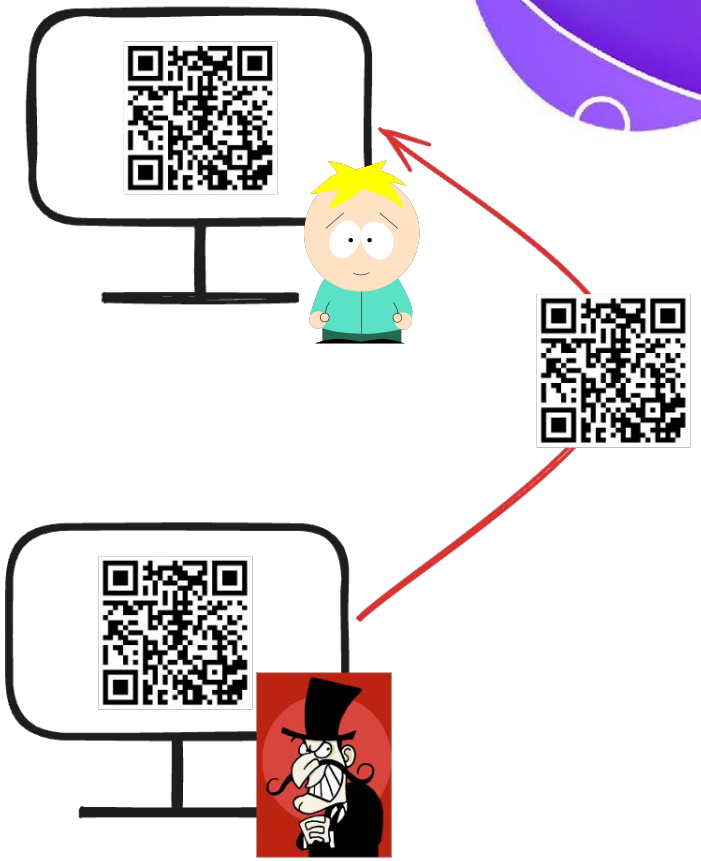


Tunnel Server

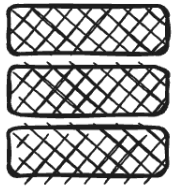


PHISHING FOR PASSKEYS

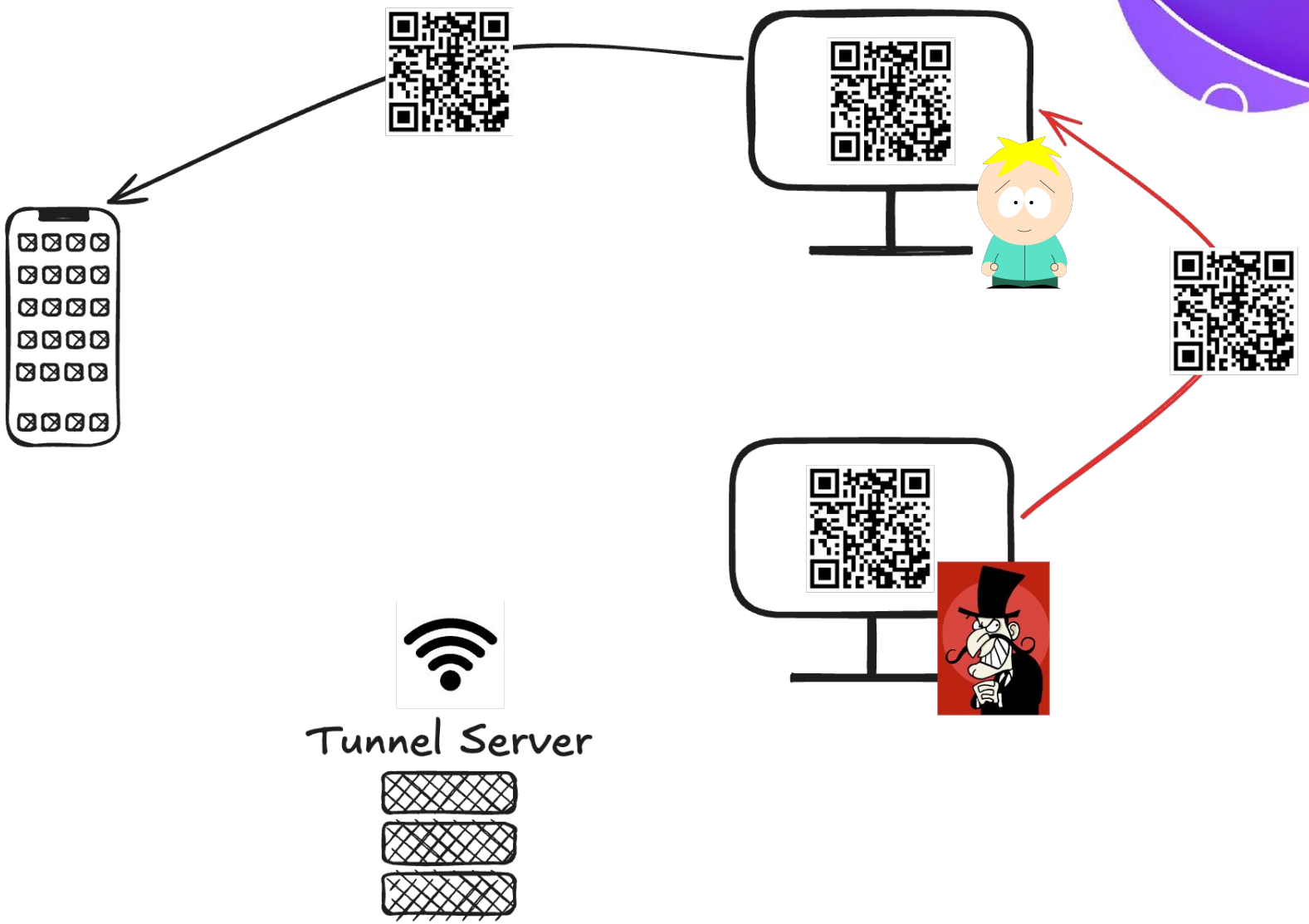


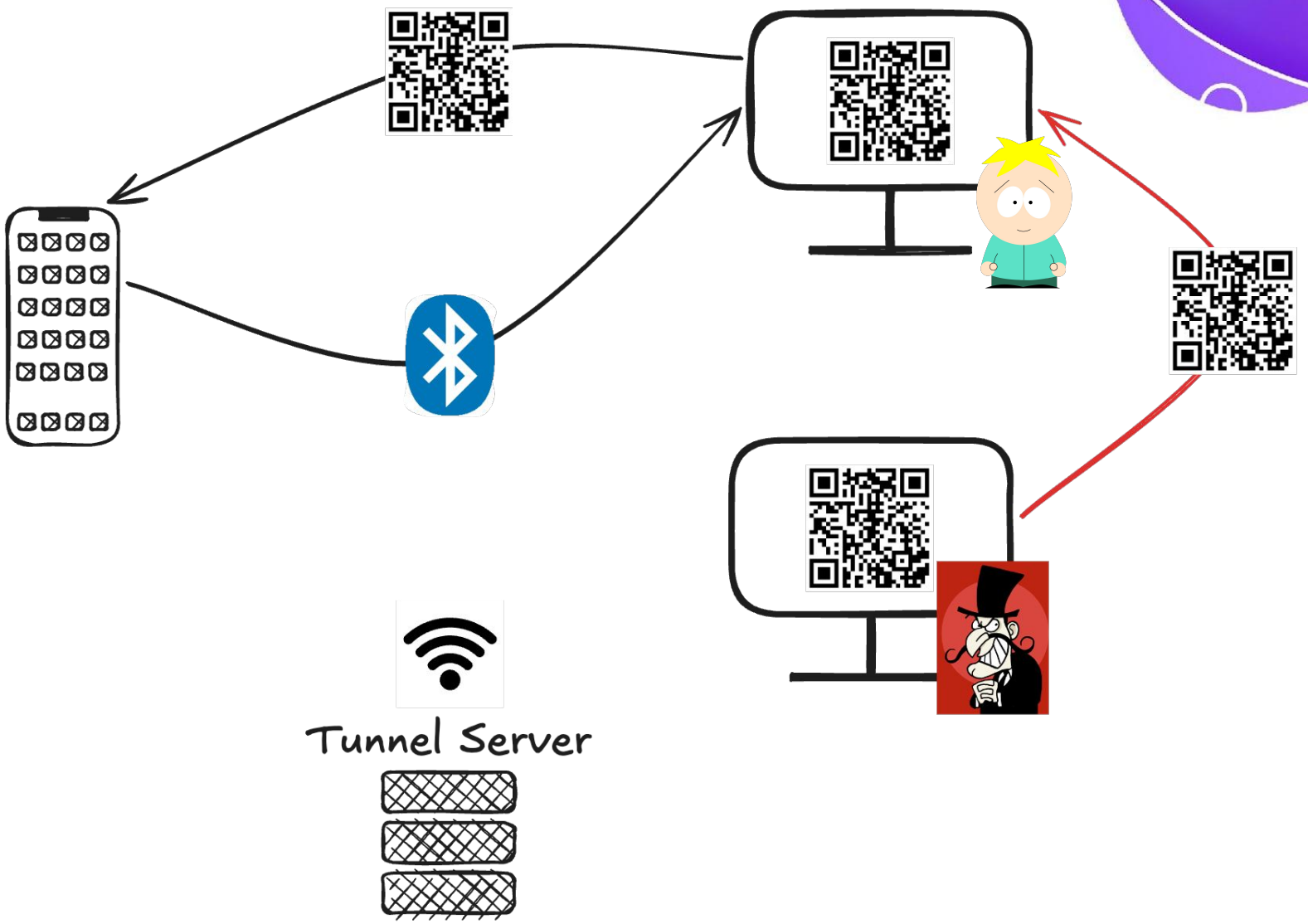


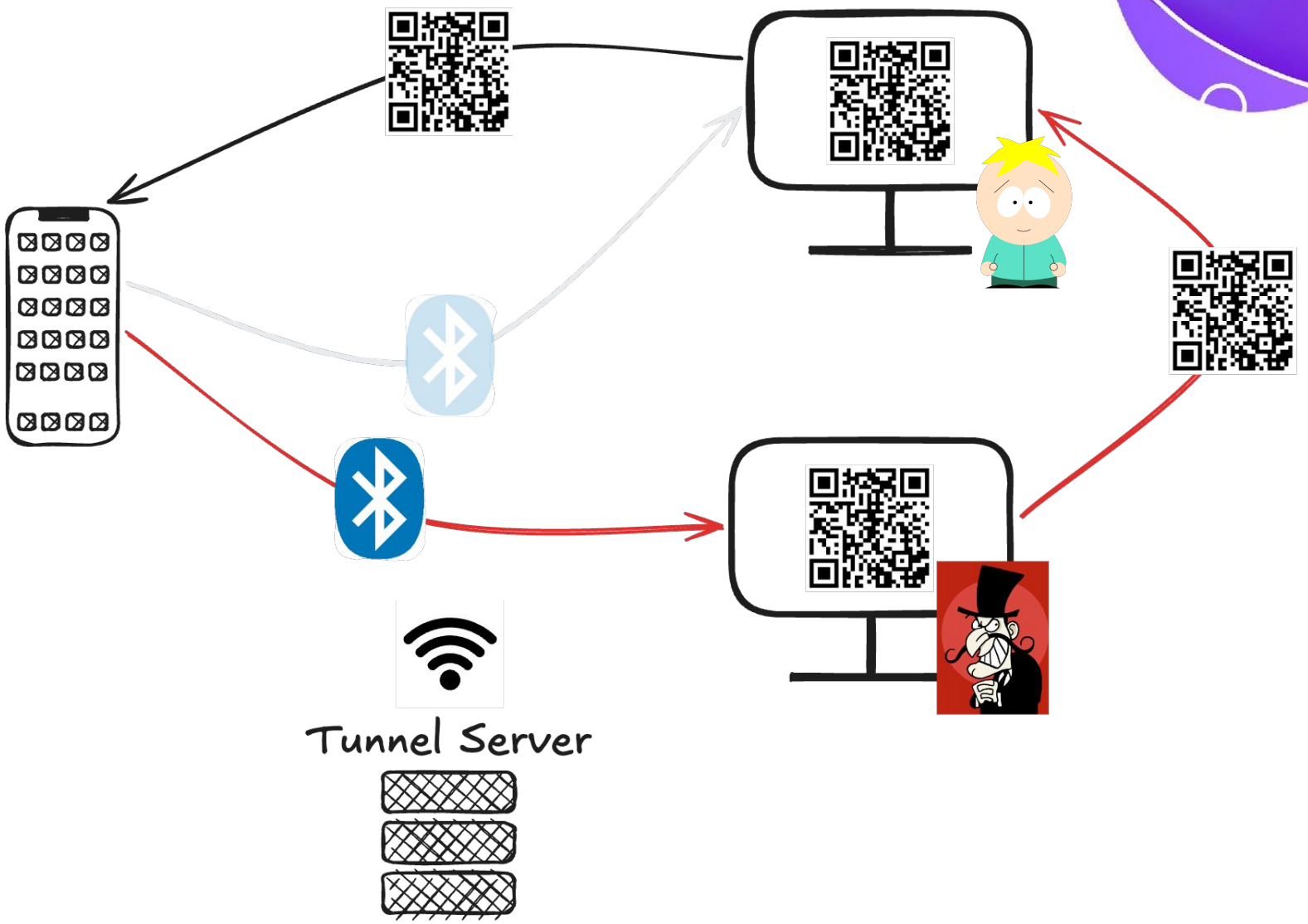
Tunnel Server

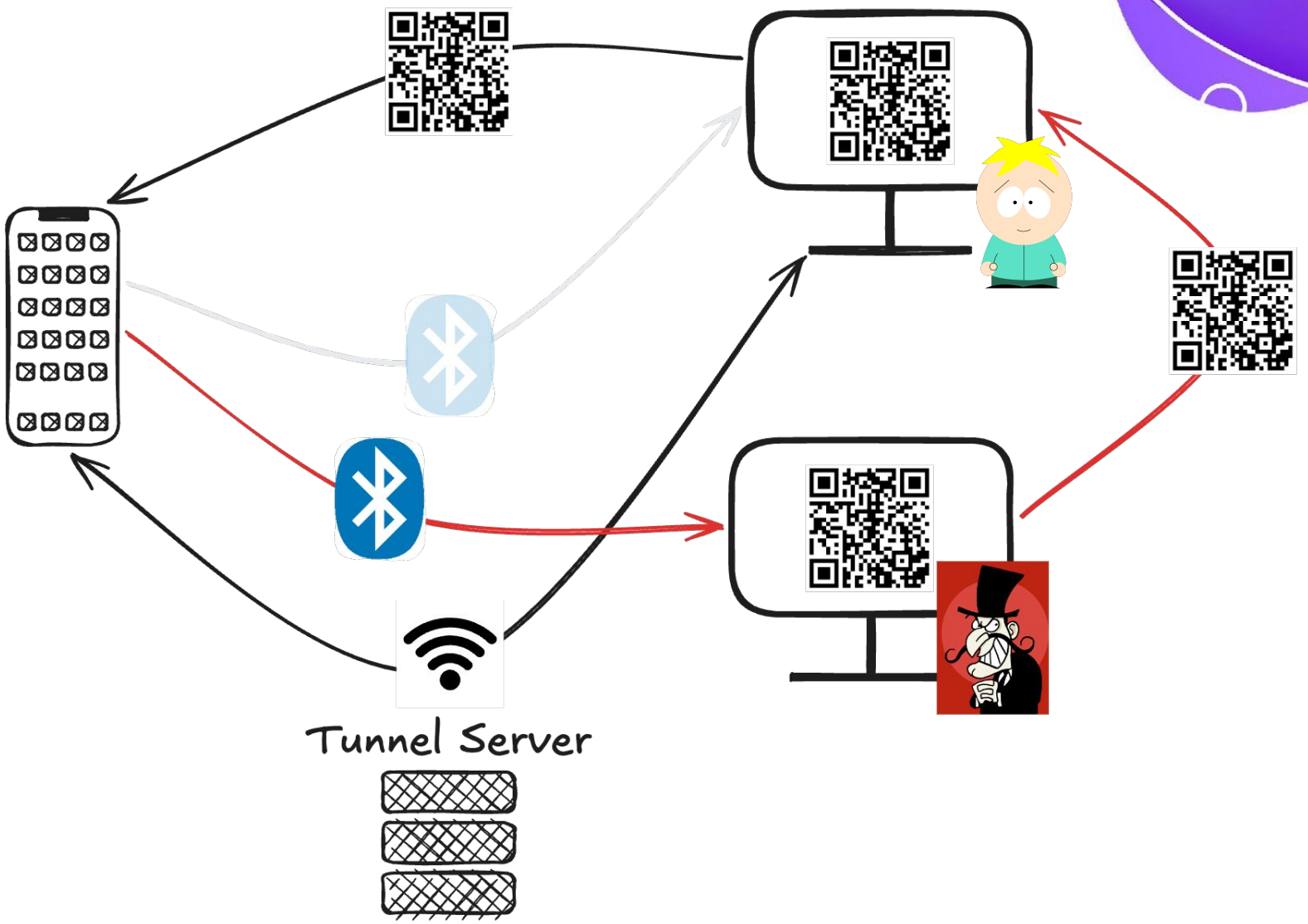


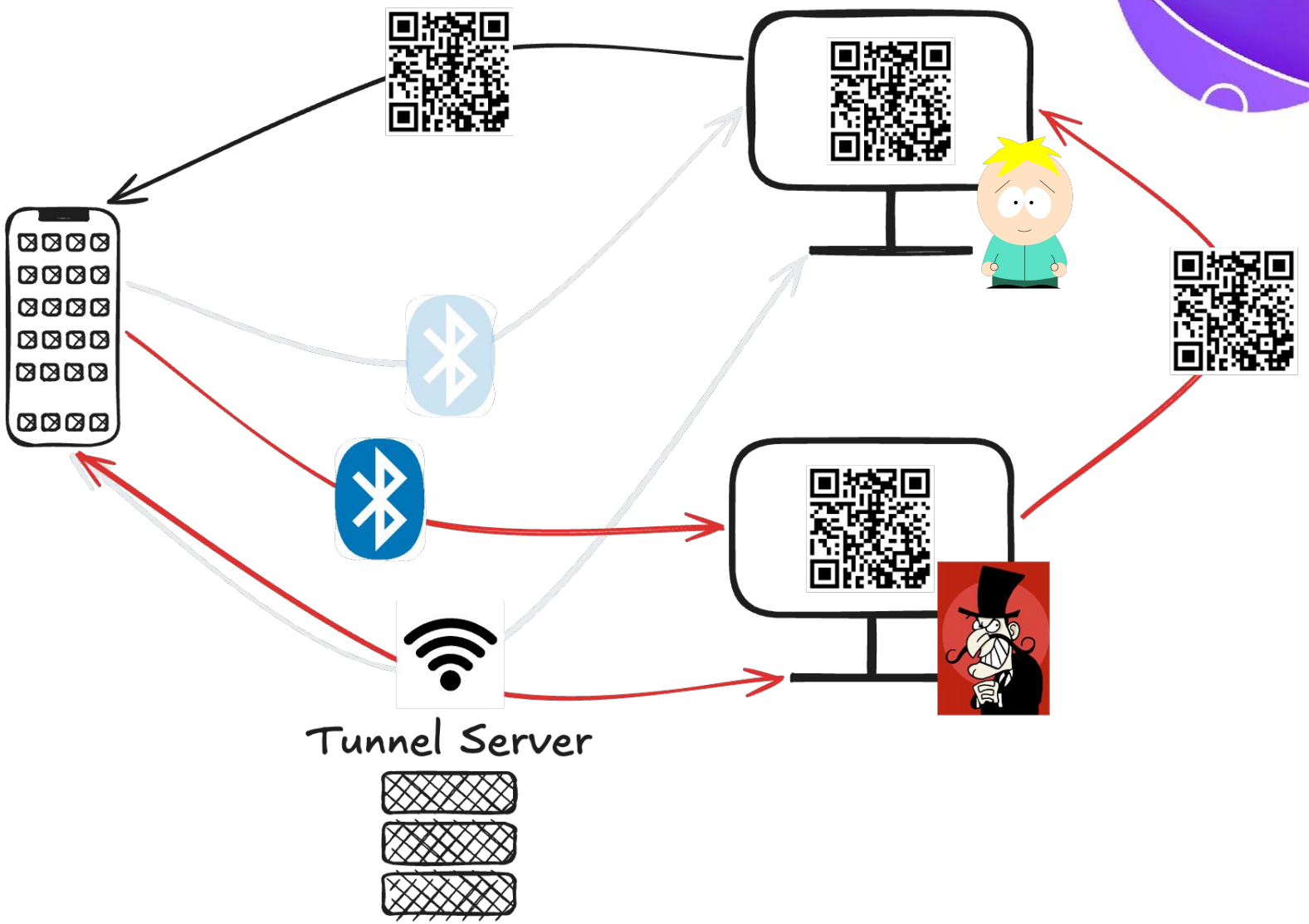
PHISHING FOR PASSKEYS



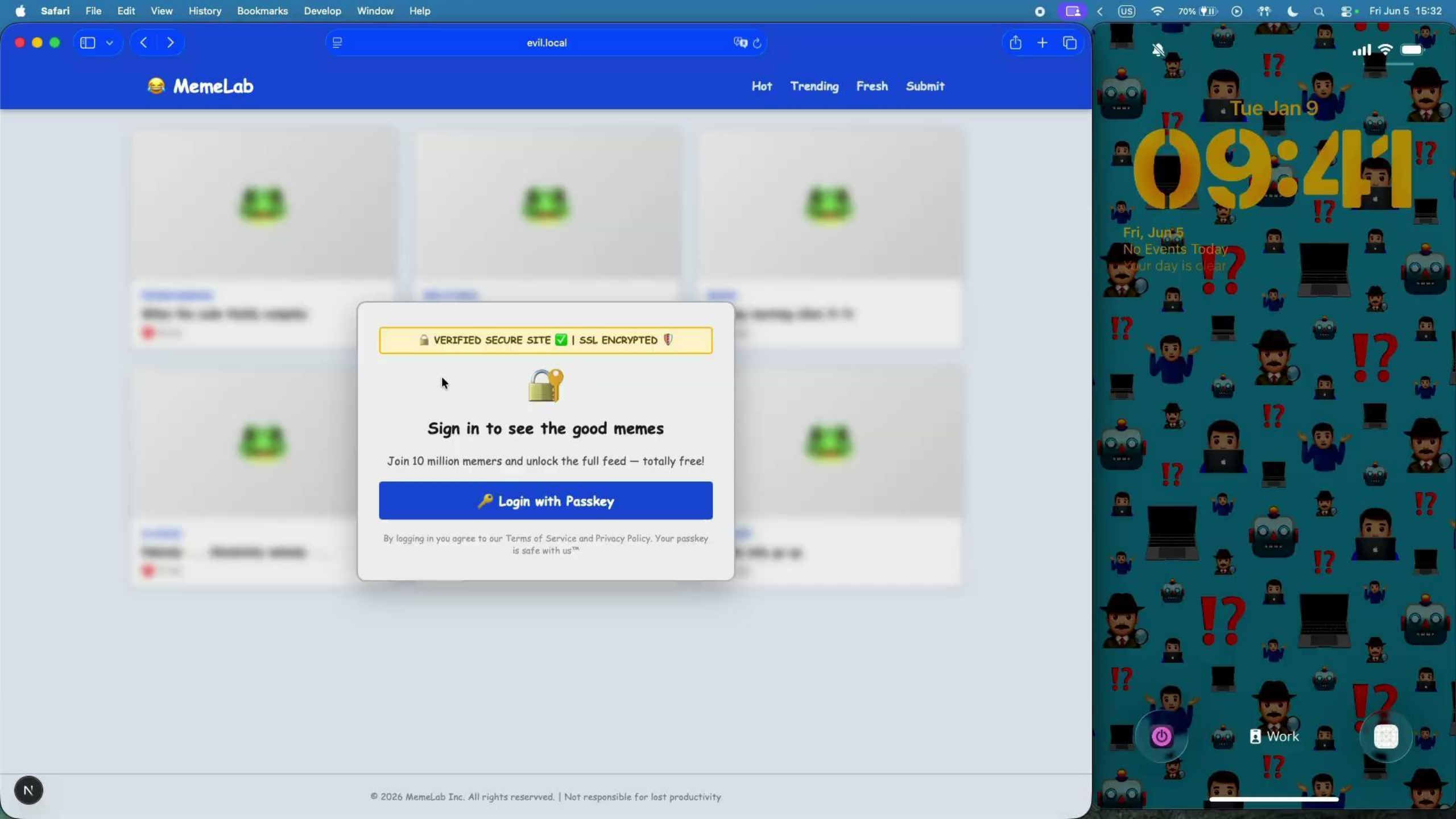

Tunnel Server

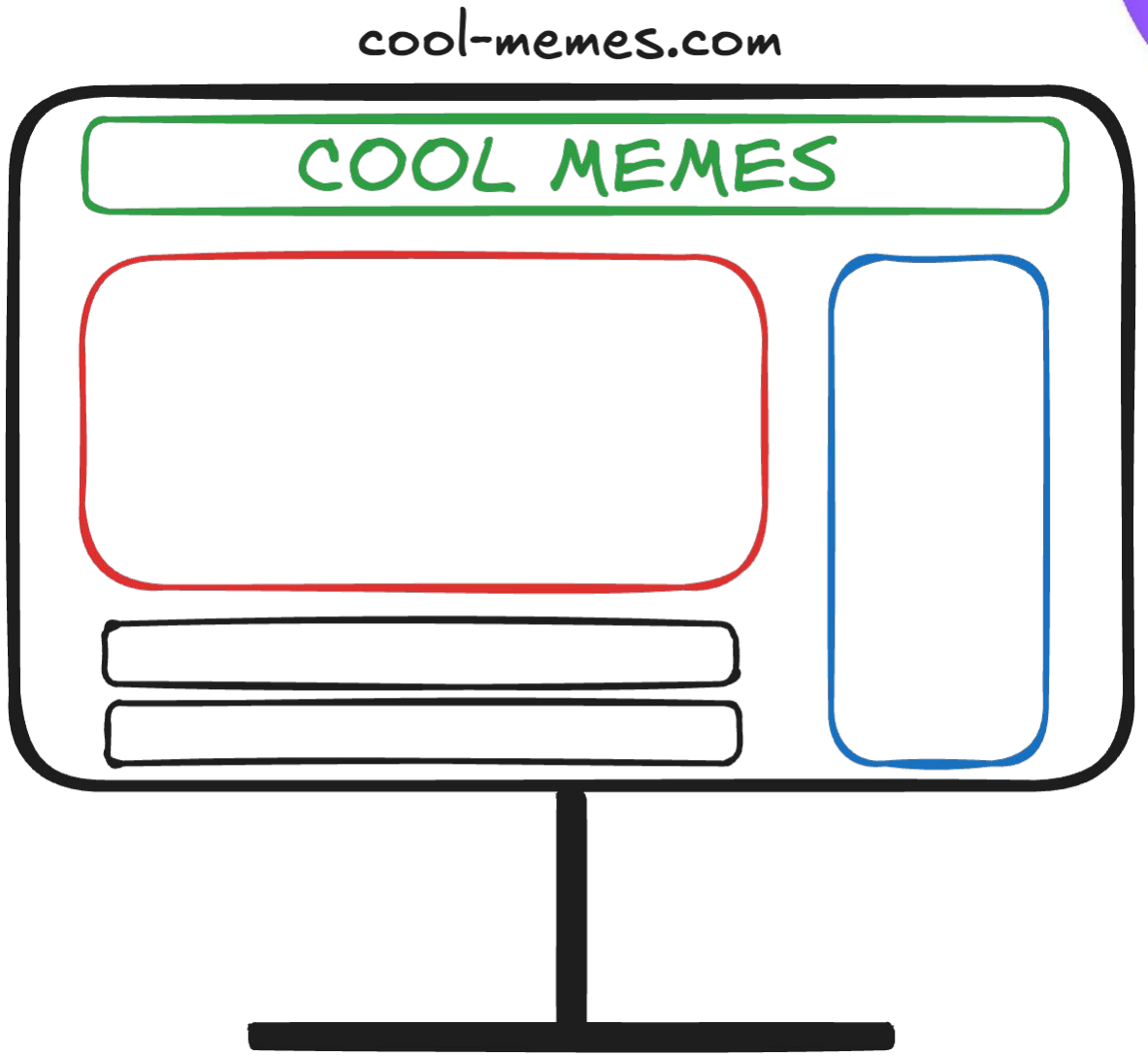

Tunnel Server

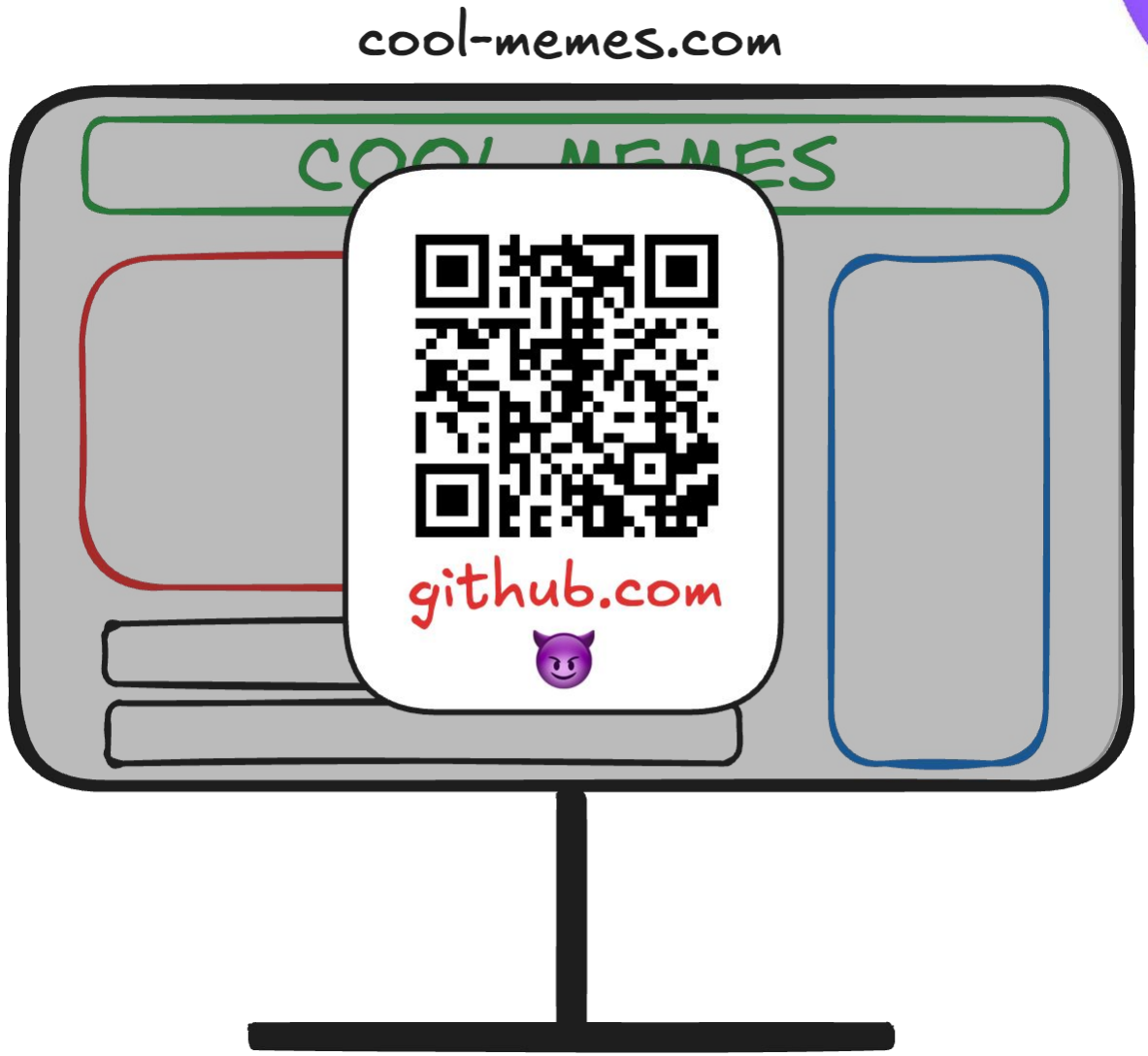

Tunnel Server


Tunnel Server

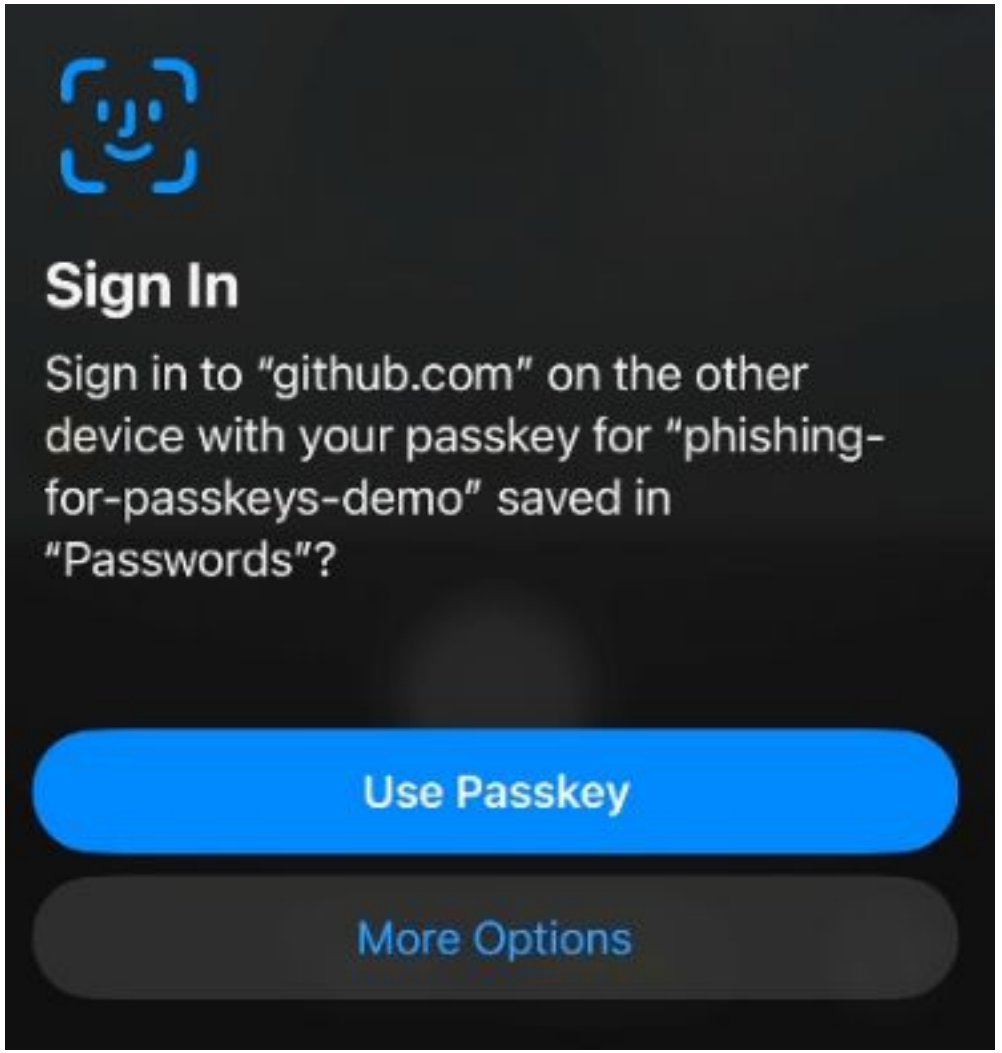
Live Demo 🤞


Safari File Edit View History Bookmarks Develop Window Help
Fri Jun 5 15:32
evil.local
MemeLab
Hot Trending Fresh Submit
VERIFIED SECURE SITE ✅ | SSL ENCRYPTED
Sign in to see the good memes
Join 10 million memers and unlock the full feed — totally free!
Login with Passkey
By logging in you agree to our Terms of Service and Privacy Policy. Your passkey is safe with us™
© 2026 MemeLab Inc. All rights reserved. | Not responsible for lost productivity
Tue Jan 9
09:41
Fri, Jun 5
No Events Today
Your day is clear
Work

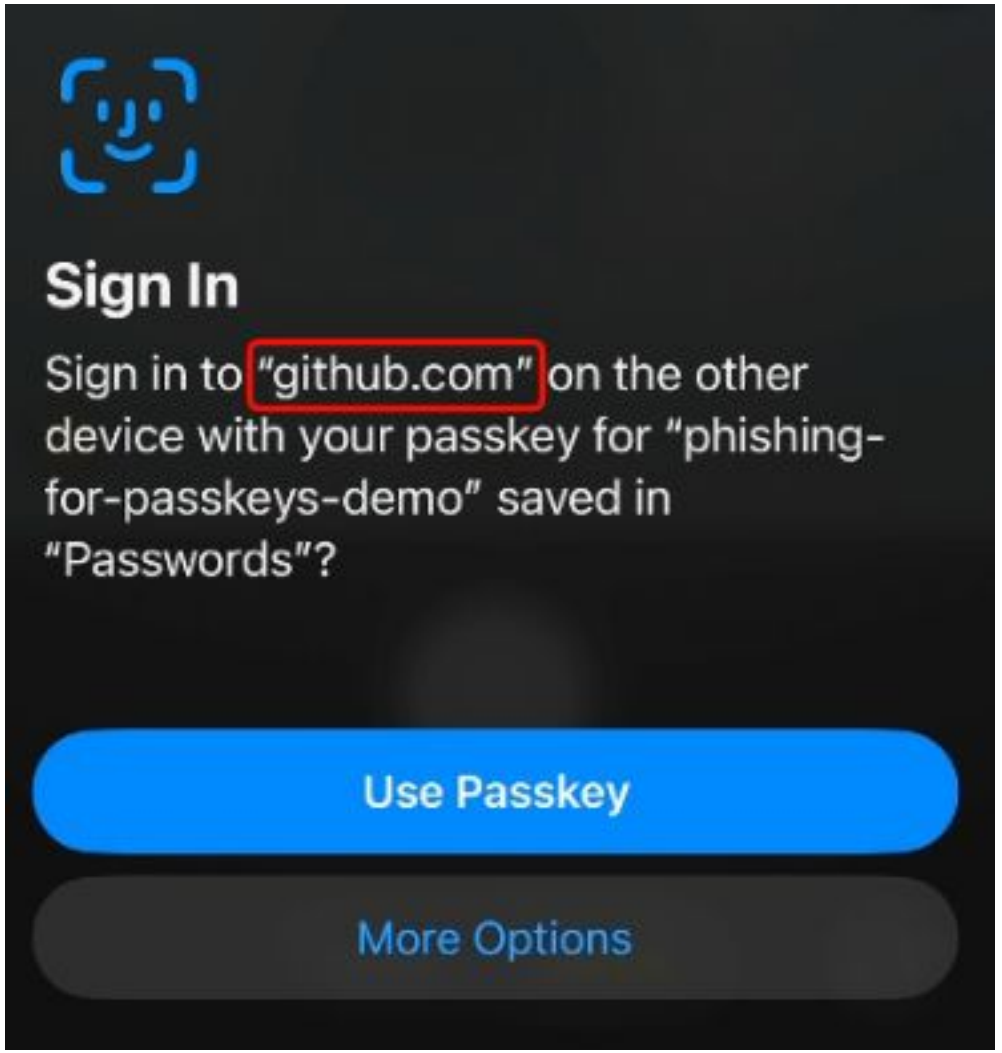

cool-memes.com
COOL MEMES


cool-memes.com
COOL MEMES
github.com

PHISHING FOR PASSKEYS



PHISHING FOR PASSKEYS

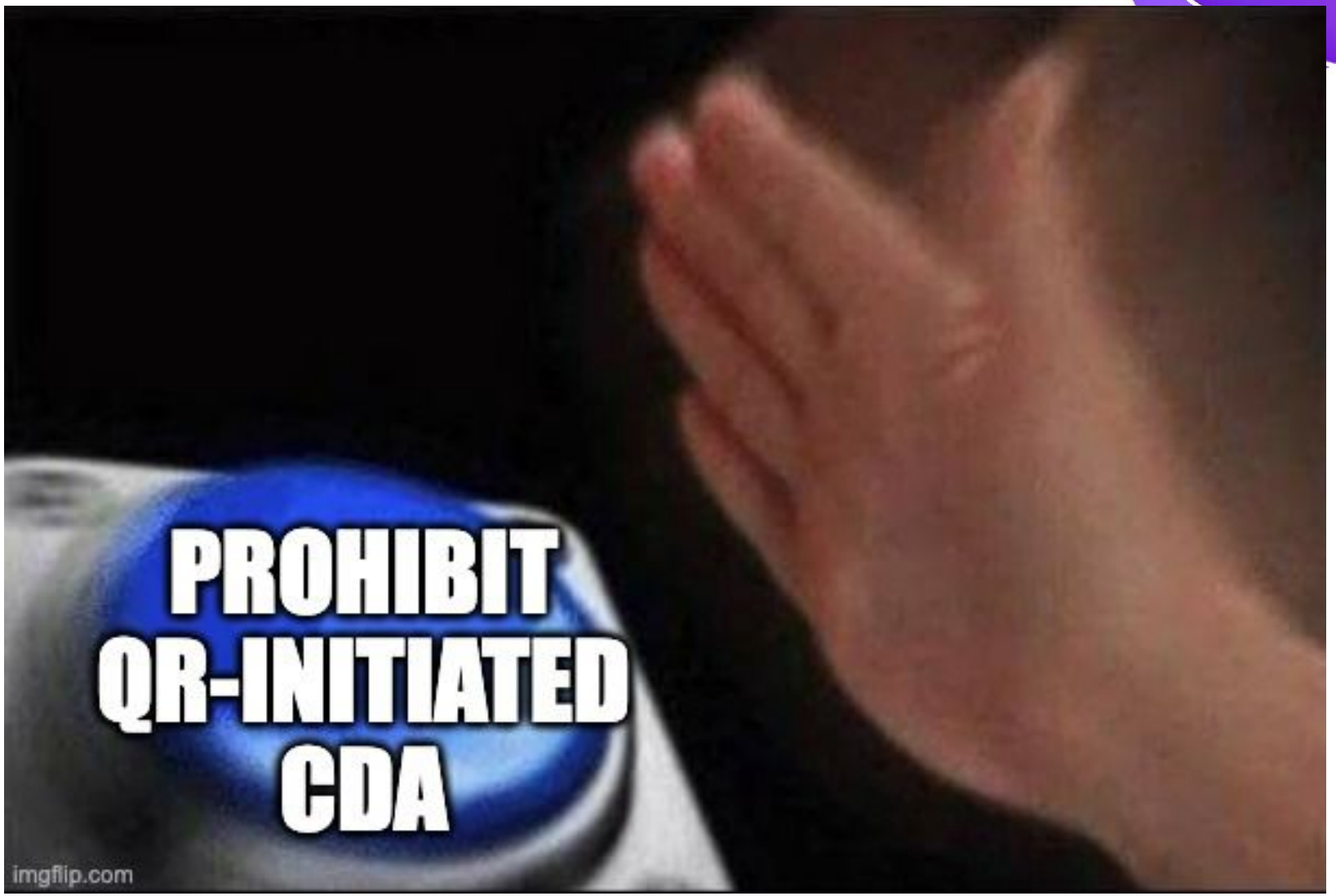


Threat Model

- Victim must be tricked into not noticing
  - Spear phishing attack & fake website
  - Fake authentication UI (usually rendered by browser, not website!)
  - Forced QR-initiated CDA
- Attack requires user interaction (open phishing page, scan QR code, complete authentication)
- Attacker must have device within BLE range of victim **while the victim's trying to authenticate**
- Successful attack → attacker gains session token

You should care if you...

- ... support passkeys for authentication
- ... expect highly motivated and technically skilled attackers
- ... expect attackers that can get within BLE range (up to ~100m) of users (e.g. parked in front of office, travelling on the same train)

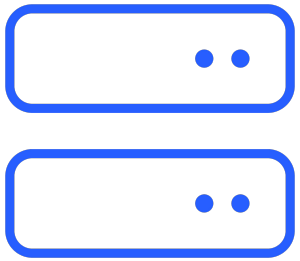

PROHIBIT
QR-INITIATED
CDA
imgflip.com

Can RPs prohibit QR-initiated CDA?

Relying Party (RP)

Client

Authenticator



Can RPs prohibit QR-initiated CDA?

Based on CTAP

- Relying party is not involved at all

→ ❌

Based on WebAuthn

→ 👀👉👈

PublicKeyCredential RequestOptions

- challenge
- timeout
- allowCredentials
  - id
  - type
  - transports 🔎⛔
- userVerification
- extensions

PublicKeyCredential

- id
- authenticatorAttachment 🔎⛔
- type
- response
  - userHandle
  - signature
  - authenticatorData
    - rpIdHash
    - flags
      - userPresent
      - ...
    - counter
    - ...
  - clientDataJSON
    - challenge
    - ...

PublicKeyCredential RequestOptions

- challenge
- timeout
- allowCredentials
  - id
  - type
  - transports
- userVerification
- extensions

PublicKeyCredential

- id
- authenticatorAttachment
- type
- response
  - userHandle
  - signature
  - authenticatorData
    - rpIdHash
    - flags
      - userPresent
      - ...
    - counter
    - ...
  - clientDataJSON
    - challenge
    - ...

PublicKeyCredential RequestOptions

- challenge
- timeout
- allowCredentials
  - id
  - type
  - transports
- userVerification
- extensions

PublicKeyCredential

- id
- authenticatorAttachment
- type
- response
  - userHandle
  - signature
  - authenticatorData
    - rpIdHash
    - flags
      - userPresent
      - ...
    - counter
    - ...
  - clientDataJSON
    - challenge
    - ...

Can RPs prohibit QR-initiated CDA?

Based on CTAP

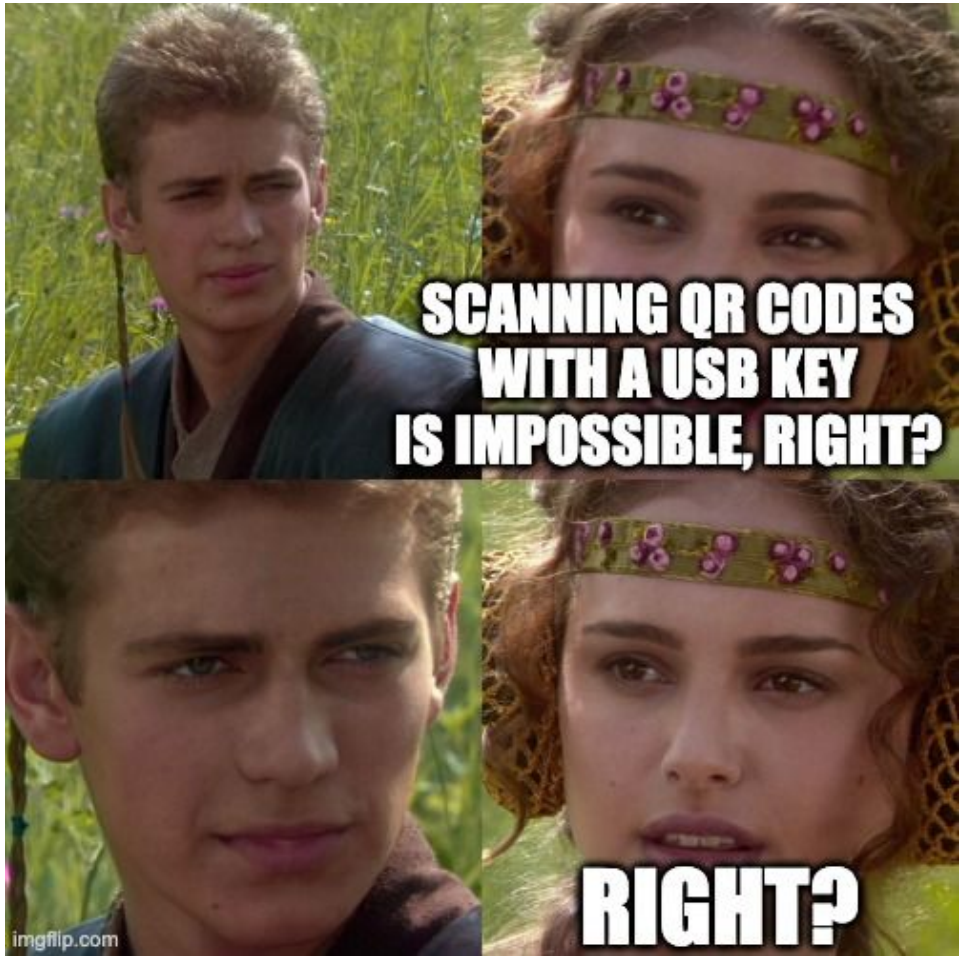
- Relying party is not involved at all

→ ❌

Based on WebAuthn

- Relying Party can ask client to not allow certain transports (e.g. BLE)
- Client specifies authenticator attachment in response (e.g. cross-platform)
- Insufficient integrity protection 🫠

→ ❌



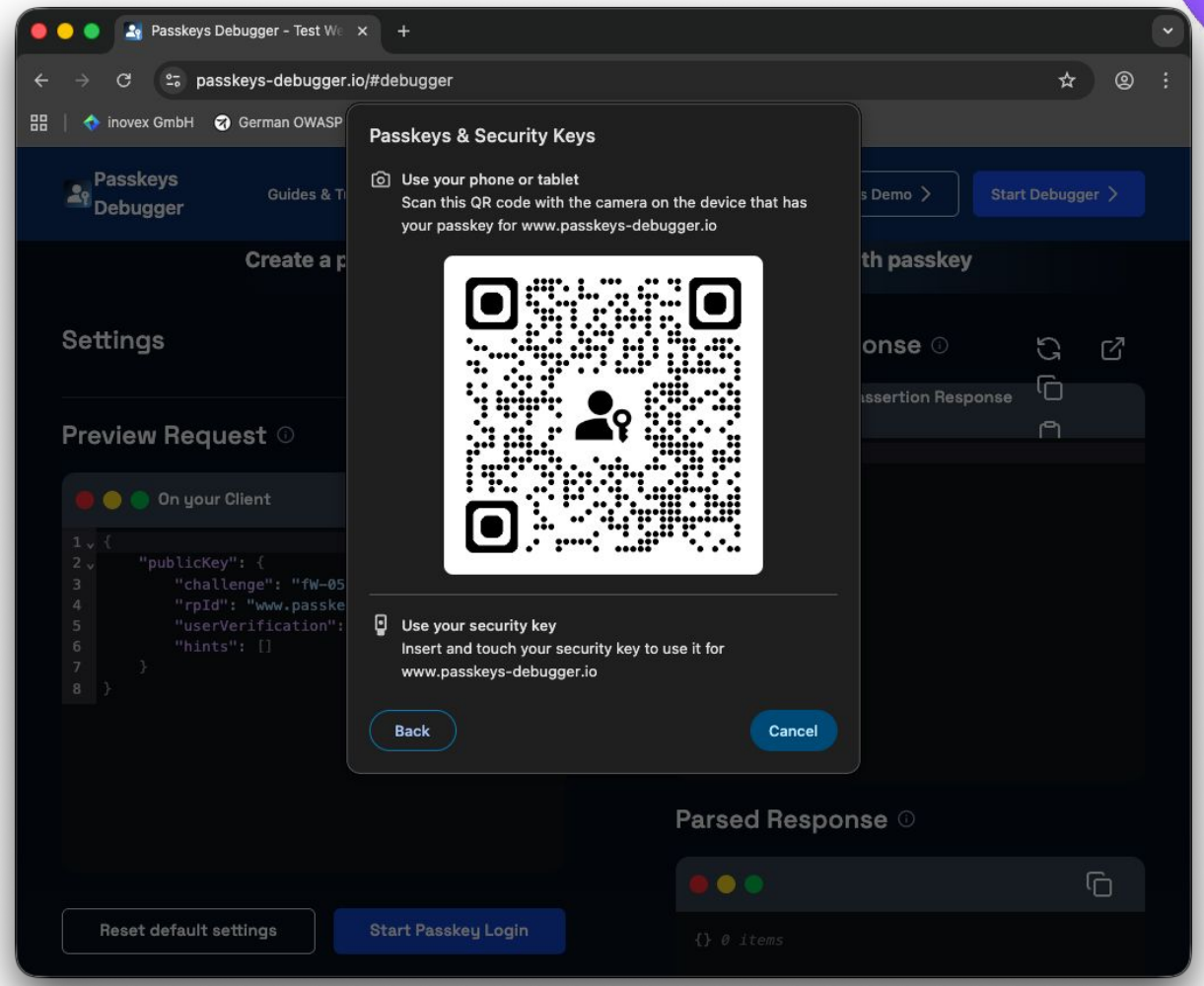
https://denniskniep.github.io/posts/14-fido-cross-device-phishing/

https://github.com/w3c/webauthn/issues/2349

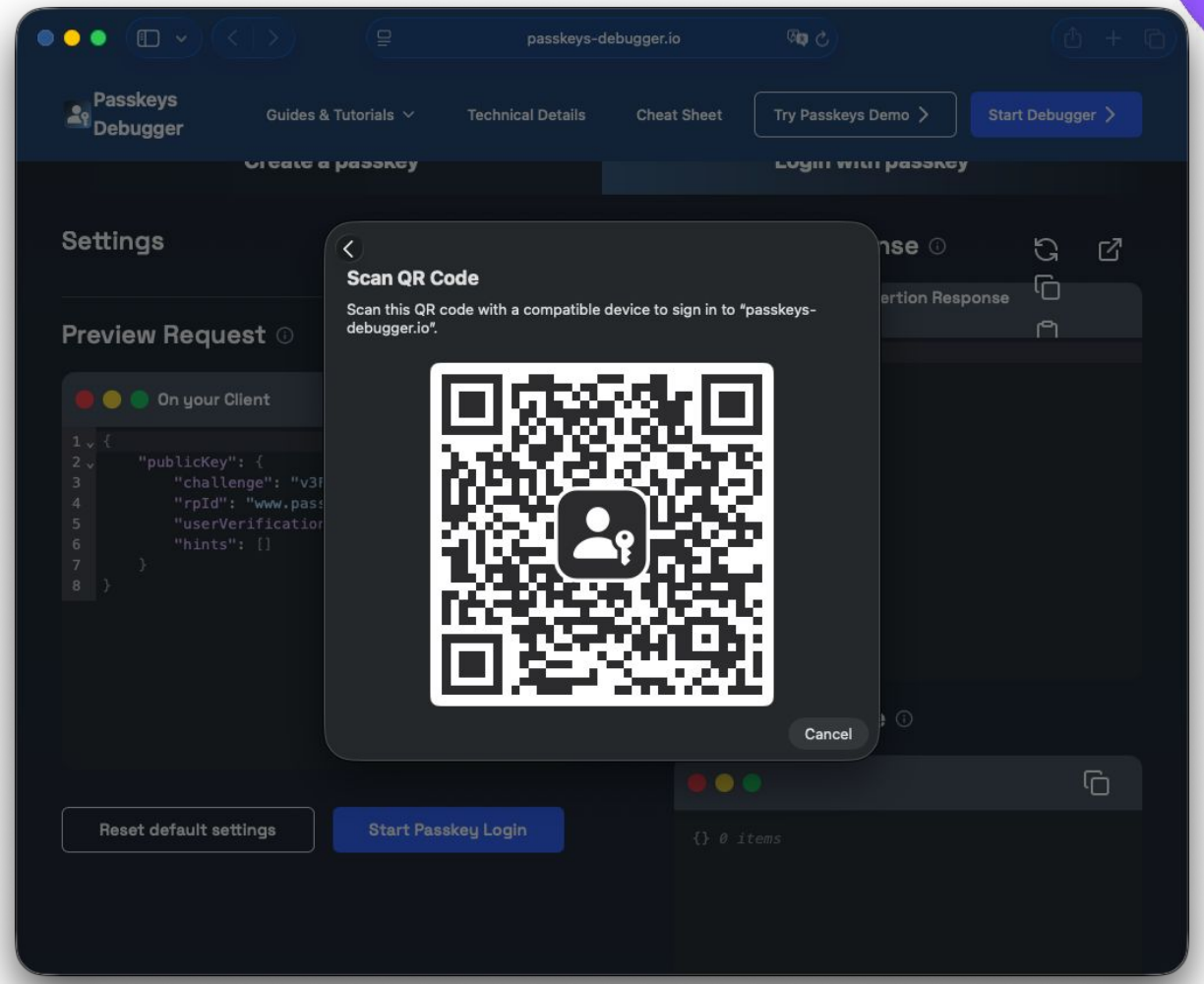
Other prevention methods

- More layers of authentication (e.g. identity managed devices)
- Improve WebAuthn & CTAP
  - Cryptographically protect properties regarding authenticator attachment & transport
  - Not possible short-term
  - W3C seems to consider this outside WebAuthn's threat model
- Educate users
  - Attackers need to force QR-initiated CDA
  - Browsers can render UI in positions unavailable to websites (outside of the website's rectangle)
  - Relying on user vigilance has proven ineffective (see phishing in general)

PHISHING FOR PASSKEYS



PHISHING FOR PASSKEYS



Other prevention methods

- More layers of authentication (e.g. identity managed devices)
- Improve WebAuthn & CTAP
  - Cryptographically protect properties regarding authenticator attachment & transport
  - Not possible short-term
  - W3C seems to consider this outside WebAuthn's threat model
- Educate users
  - Attackers need to force QR-initiated CDA
  - Browsers can render UI in positions unavailable to websites (outside of the website's rectangle)
  - Relying on user vigilance has proven ineffective (see phishing in general)

Conclusion

- Passkeys are still resistant to regular phishing
- Depending on threat model, passkeys may be vulnerable to spear phishing attacks
- Phishing site can imitate other websites than its target
- QR-initiated CDA cannot be prohibited directly by relying parties
- Protection measures are more involved (e.g. adding more authentication layers)

OWASP GLOBAL AppSec

VIENNA'26 JUN 25-26

25 years of open source security

THANK YOU!

Michael Kuckuk

Fullstack Developer @ inovex

@michael-david-kuckuk

@LBBO@mastodontech.de

@LBBO.de

[Read the full blog post for more details!]

New Tab

Target Website

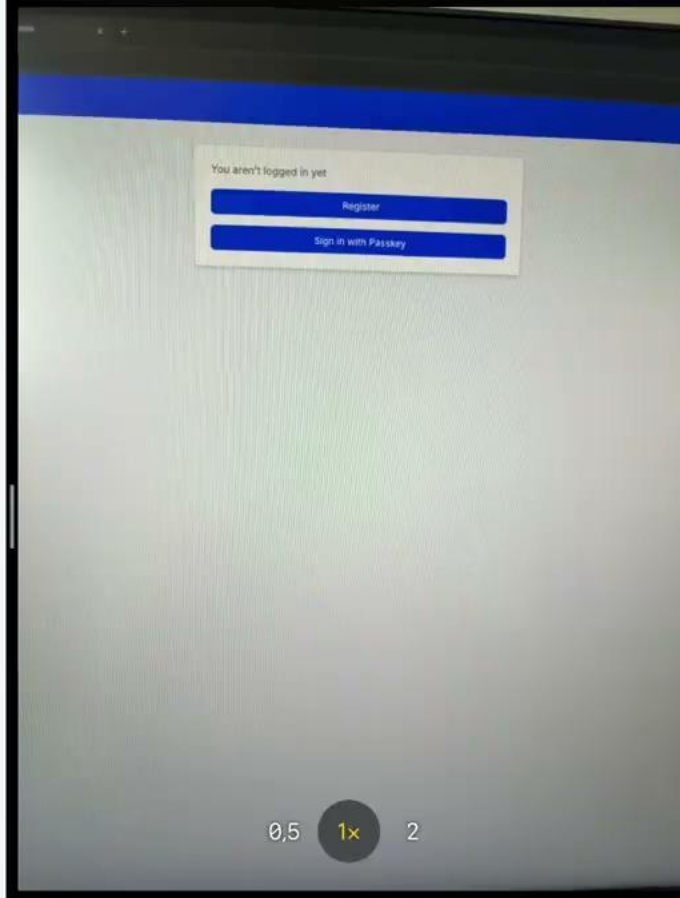
phishing-target.local:3000/demo

Target Website

You aren't logged in yet

Register

Sign in with Passkey



Chrome File Edit View History Bookmarks Profiles Tab Window Help

Wed May 20 16:13

```
mise run create-tmux
/phishing-for-passkeys main 24.15.0 16:13:18
mise run attack
[attack] $ deno run --sloppy-imports --allow-env --env-file=.env.development --allow-read --al…
[16:13:28.329] INFO (#17): Press enter to start the attack
Press enter to continue...
```

0 Chrome 1 DB 2 Apps - 3 Attack * 2026-05-20 16:13

New Tab | Target Website

phishing-target.local:3000

Target Website

You aren't logged in yet

Register

Sign in with Passkey

THIS WINDOW IS CONTROLLED BY:

Image: J.J. at the English-language Wikipedia

evil.local

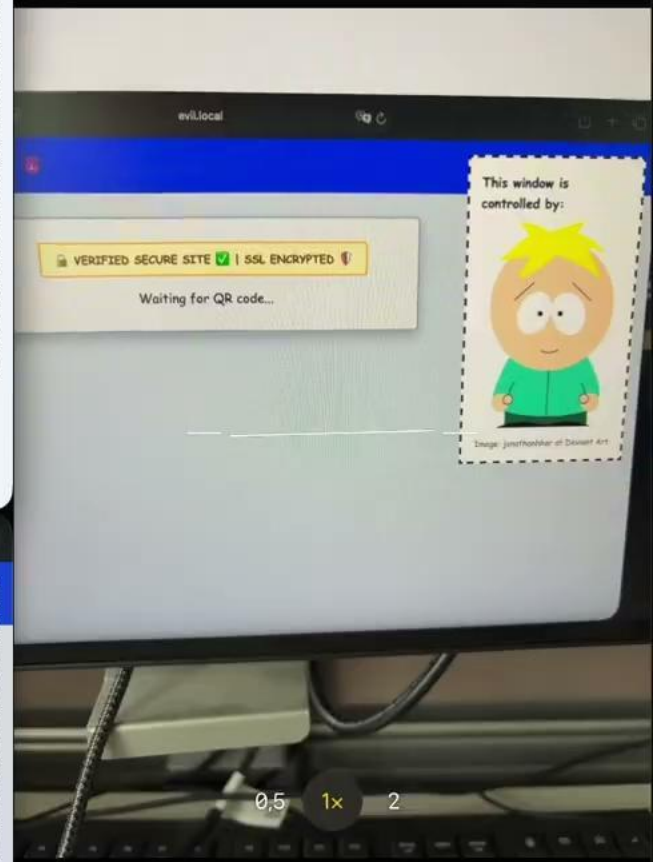
Definitely the target website 😈

🔒 VERIFIED SECURE SITE ✅ | SSL ENCRYPTED 🛡️

Waiting for QR code...

This window is controlled by:

Image: jonathanhher at Deviant Art



Ln 4, Col 1 Spaces: 2 UTF-8 LF { } TypeScript JSX