

Introducing Passkeys

Strategies and Challenges
for Developers



Clemens Hübner
German OWASP Day 2025



Clemens Hübner

Tech Lead Software Security
Engineer, Consultant, Trainer

Secure Coding, Threat Modeling, AppSec, DevSecOps



/clemens-huebner



clemens.huebner@inovex.de



@clemens@infosec.exchange



@inovexlife

blog.inovex.de

Status quo of authentication

The image shows a login form with a light gray header. In the header, there is a "Log In" link and a blue "Sign Up" button. Below the header, there is a white box containing two input fields: "Username" and "Password". Below the "Password" field is a link that says "Forgot your username or password?". At the bottom of the white box is a blue "Log In" button.

Log In **Sign Up**

Username

Password

[Forgot your username or password?](#)

Log In

User's mistakes with passwords

- Weak passwords
- Very weak passwords
- Wrong handling of passwords
- Reuse of passwords



User's mistakes with passwords



- Weak passwords → Brute Force Attacks
try a lot of passwords for a user
- Very weak passwords
- Wrong handling of passwords
- Reuse of passwords

User's mistakes with passwords



- Weak passwords → Brute Force Attacks
try a lot of passwords for a user
- Very weak passwords → Password Spraying
try weak password for multiple users
- Wrong handling of passwords
- Reuse of passwords

User's mistakes with passwords

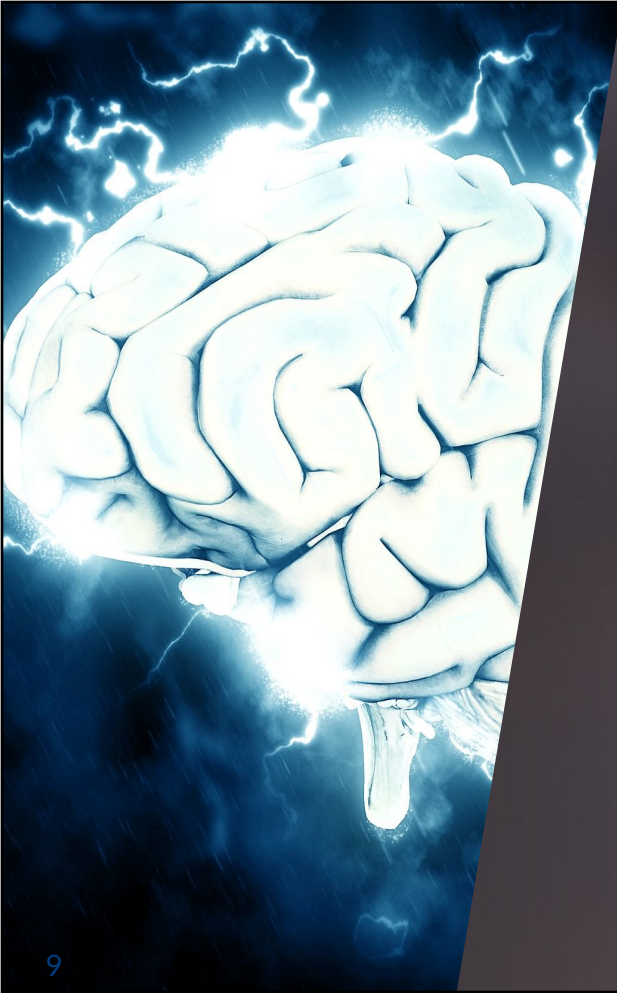


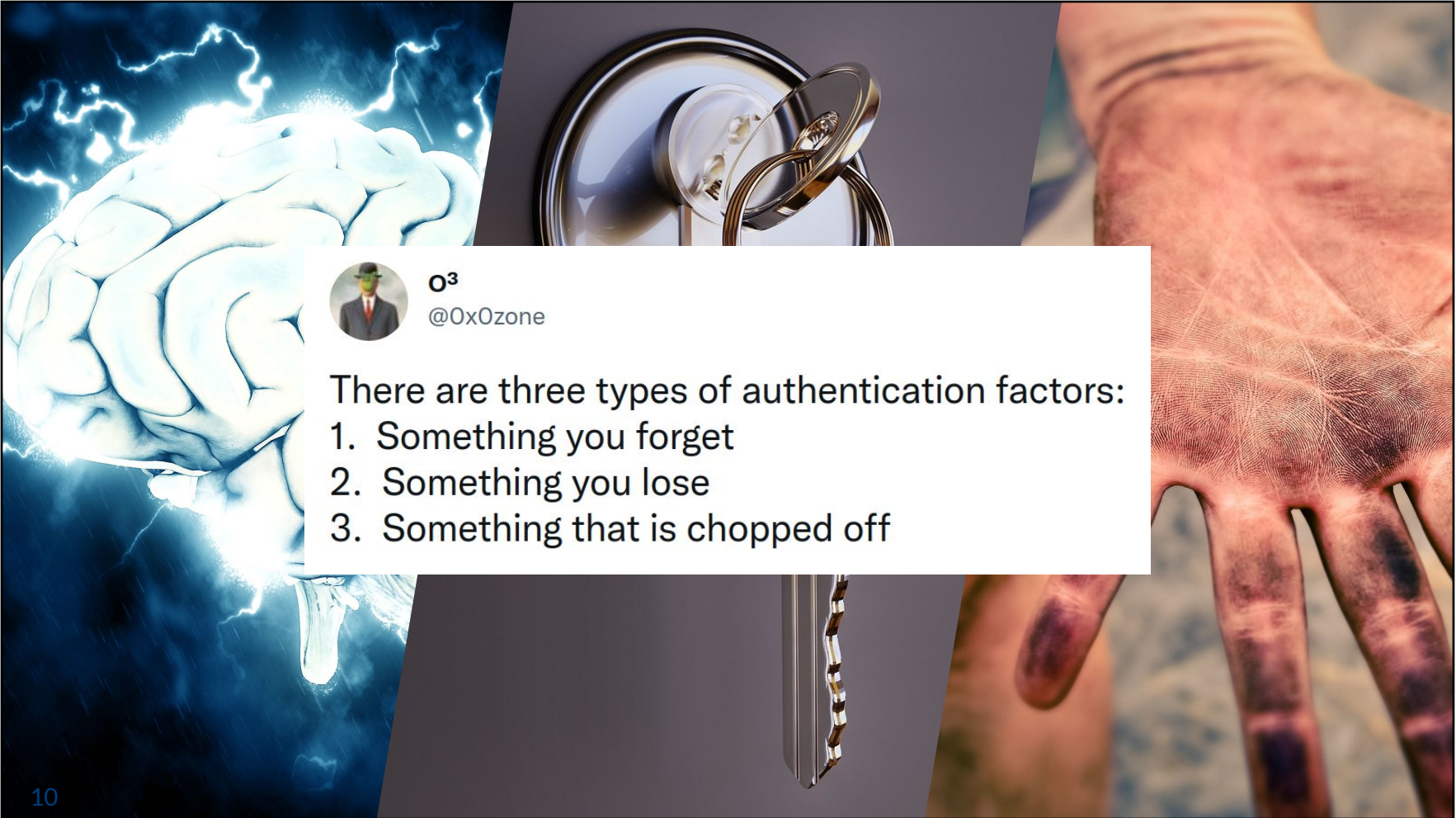
- Weak passwords → Brute Force Attacks
try a lot of passwords for a user
- Very weak passwords → Password Spraying
try weak password for multiple users
- Wrong handling of passwords → Phishing Attacks
trick the user to tell you the password
- Reuse of passwords

User's mistakes with passwords



- Weak passwords → Brute Force Attacks
try a lot of passwords for a user
- Very weak passwords → Password Spraying
try weak password for multiple users
- Wrong handling of passwords → Phishing Attacks
trick the user to tell you the password
- Reuse of passwords → Credential Stuffing
try a lot of known pairs of
username/password





o³

@OxOzone

There are three types of authentication factors:

1. Something you forget
2. Something you lose
3. Something that is chopped off

Solution: Less knowledge-based authentication

- use possessed or biometric factors
- use public-key based challenge-response (no leakage of any secret)
- strong scoping of credentials for phishing protection

→ Passkeys to the rescue



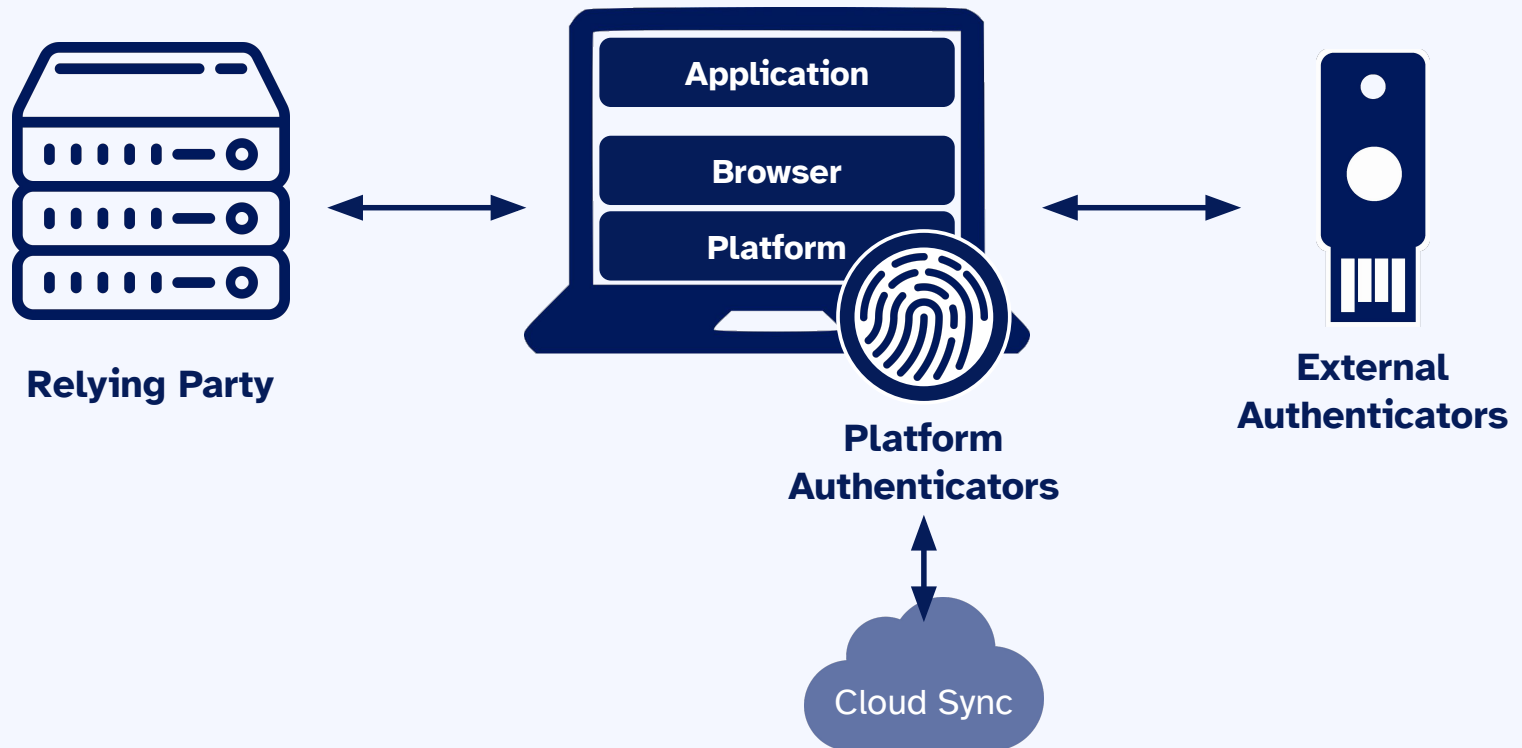


PASSKEYS

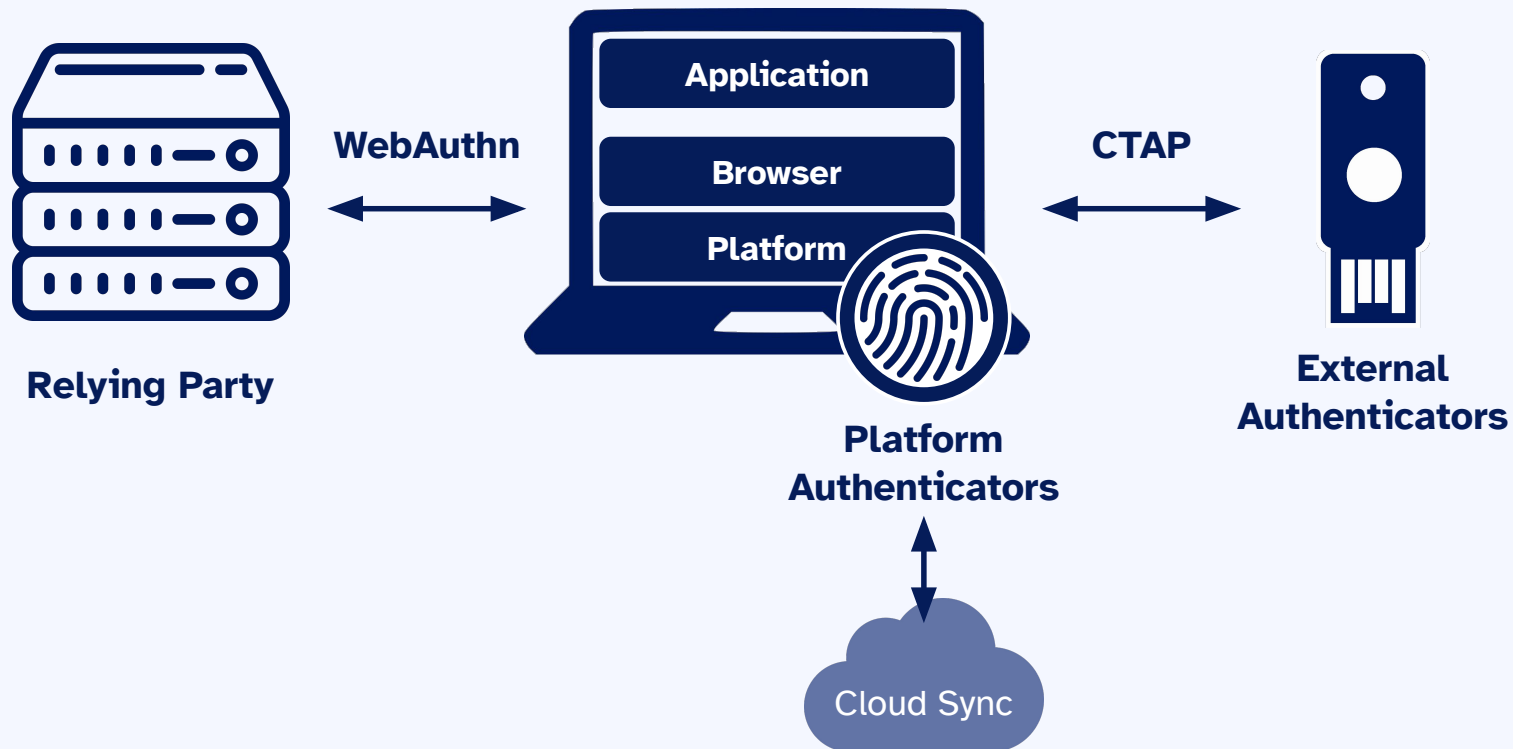
DEVS

WEBAUTHN

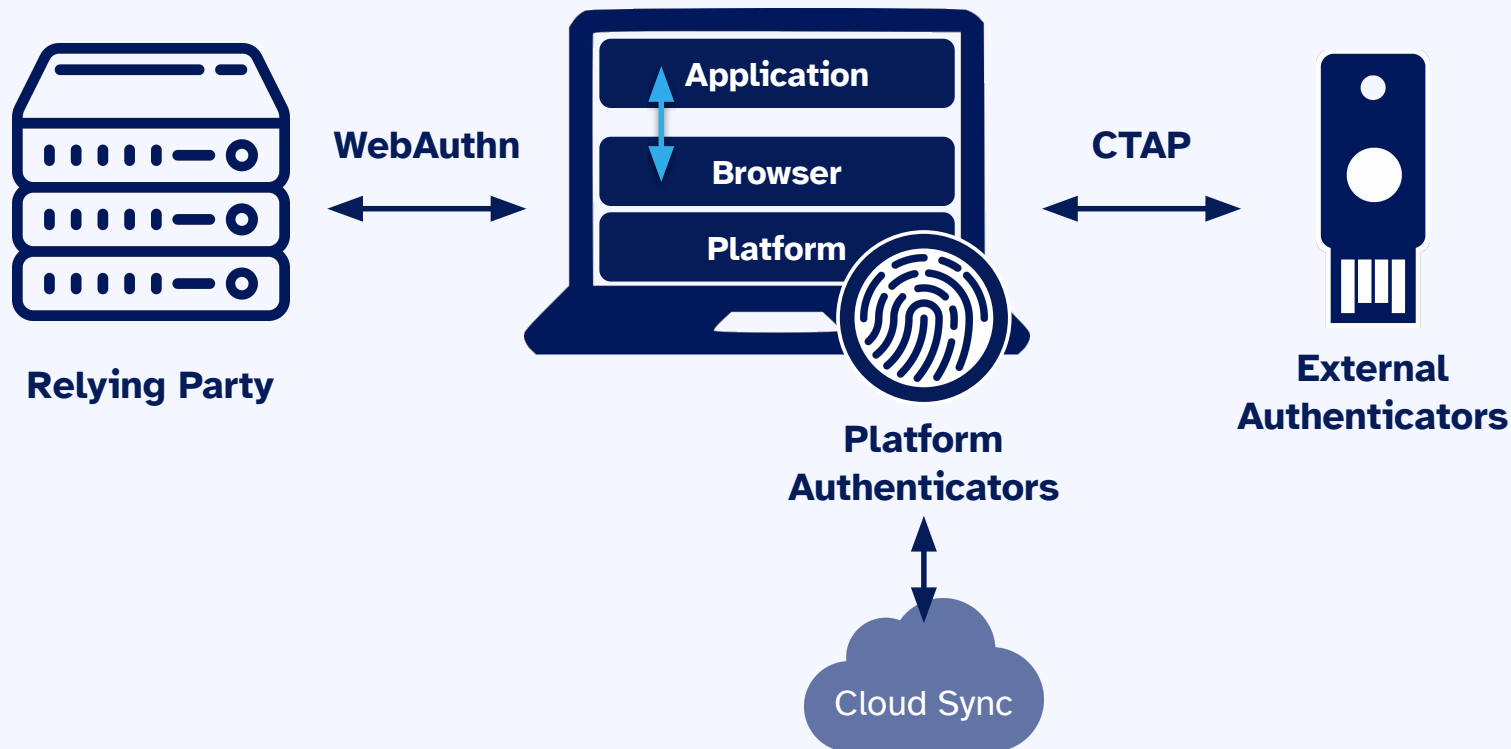
Architecture of Passkeys



Architecture of Passkeys



Architecture of Passkeys



The two Passkey Processes aka Ceremonies

Registration Ceremony

Creating a public key credential, scoped to the Relying Party, based on a user's identifier

Authentication Ceremony

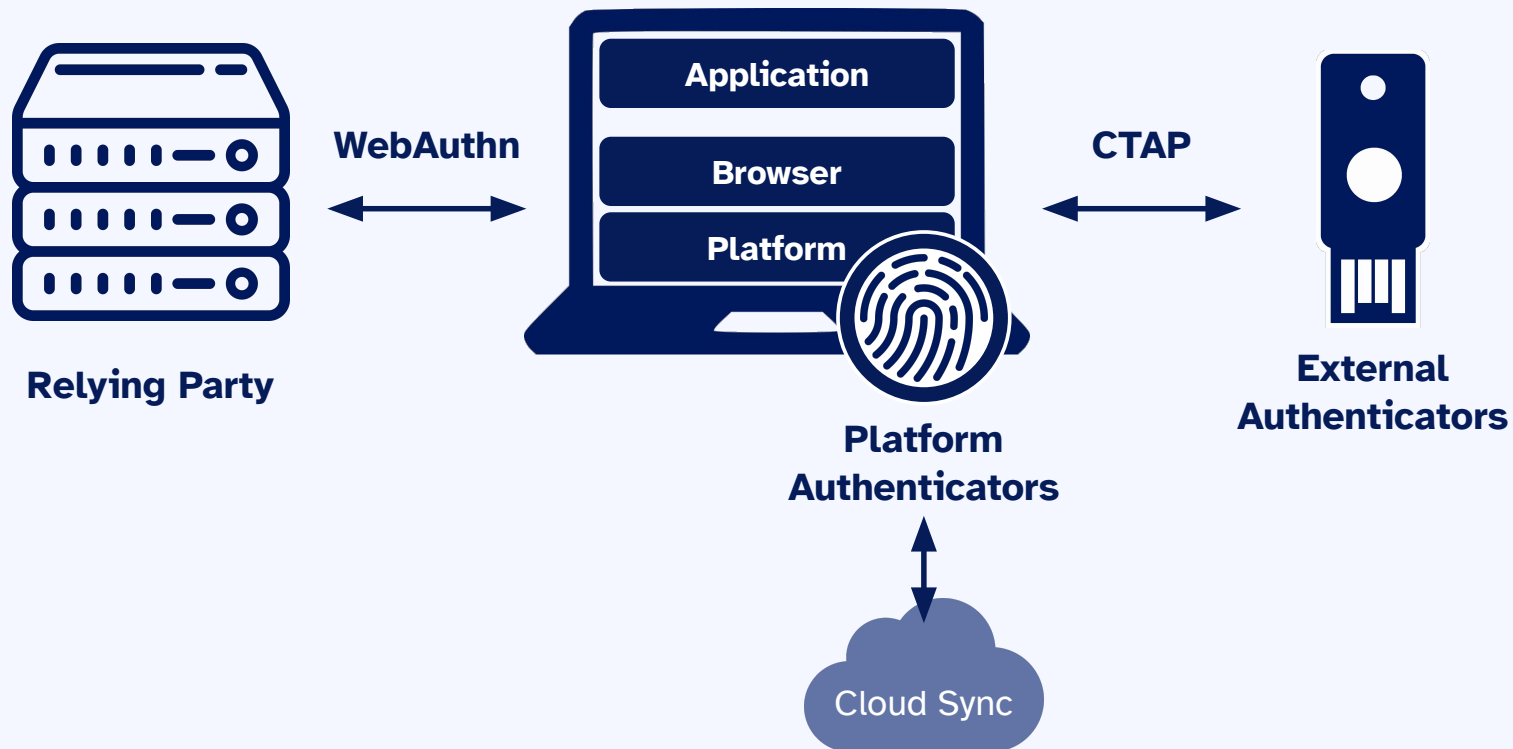
Proving the presence of the private key and the consent of the user that registered it



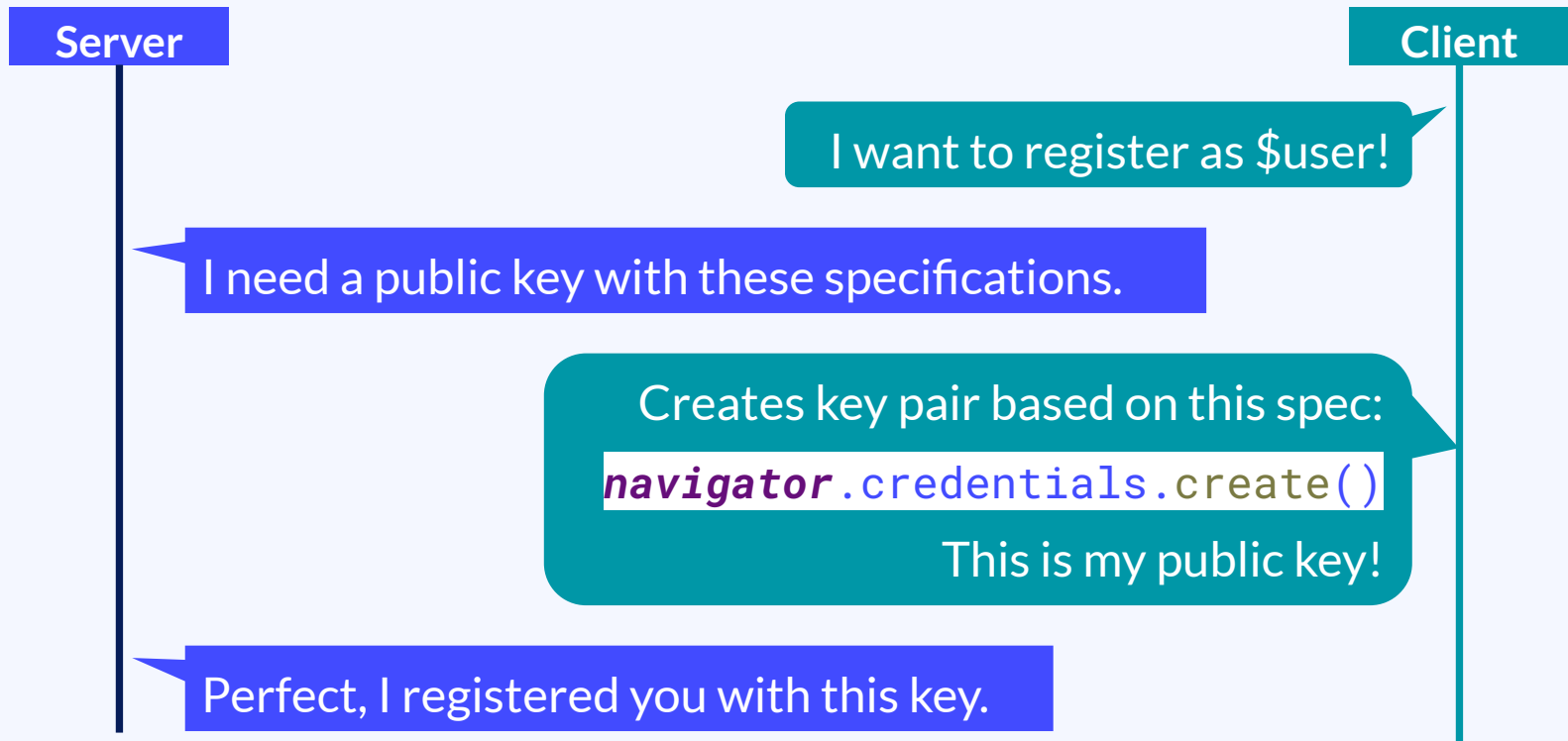
Attention: That's all!

Demo time!

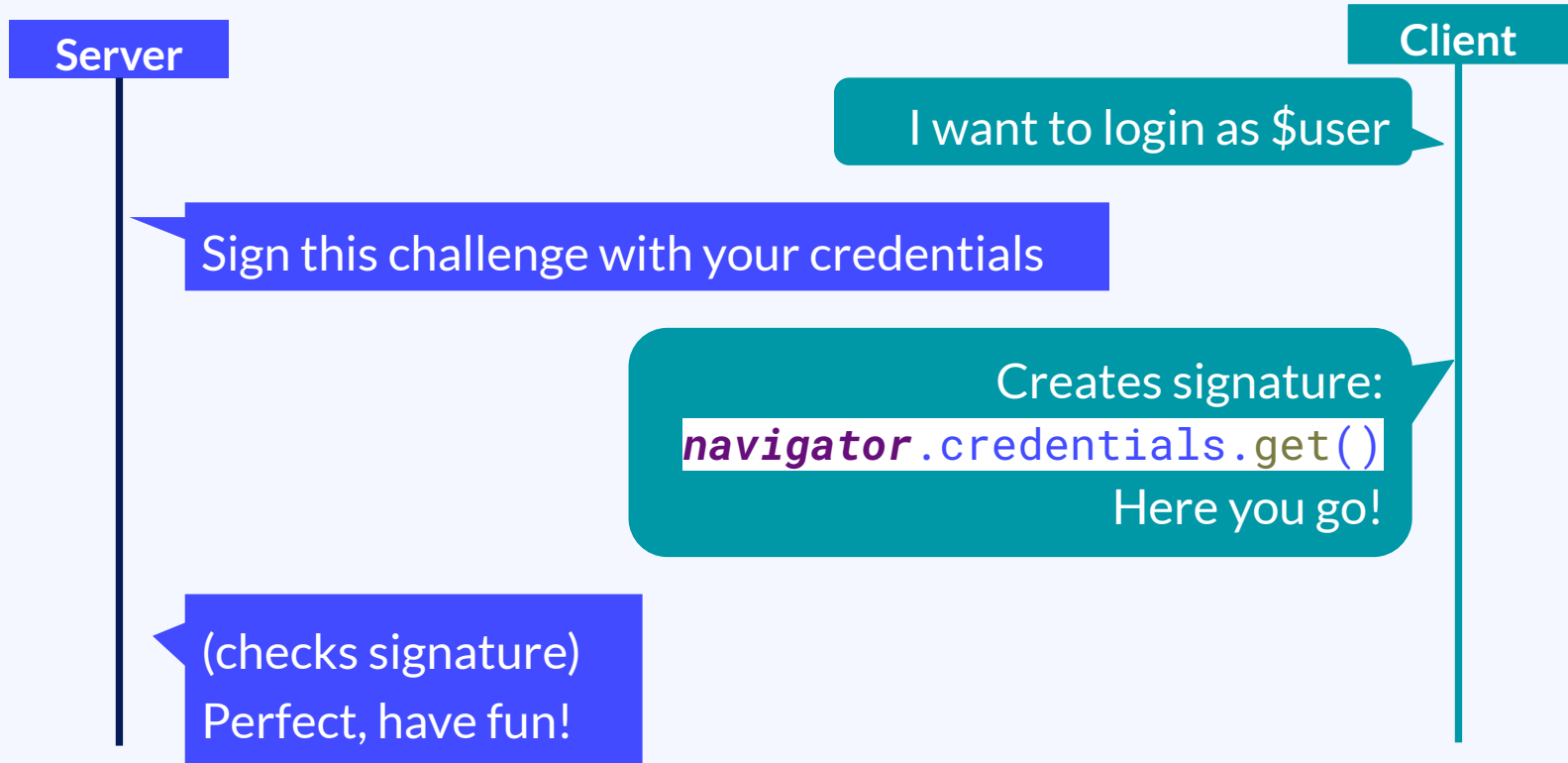
Architecture of Passkeys



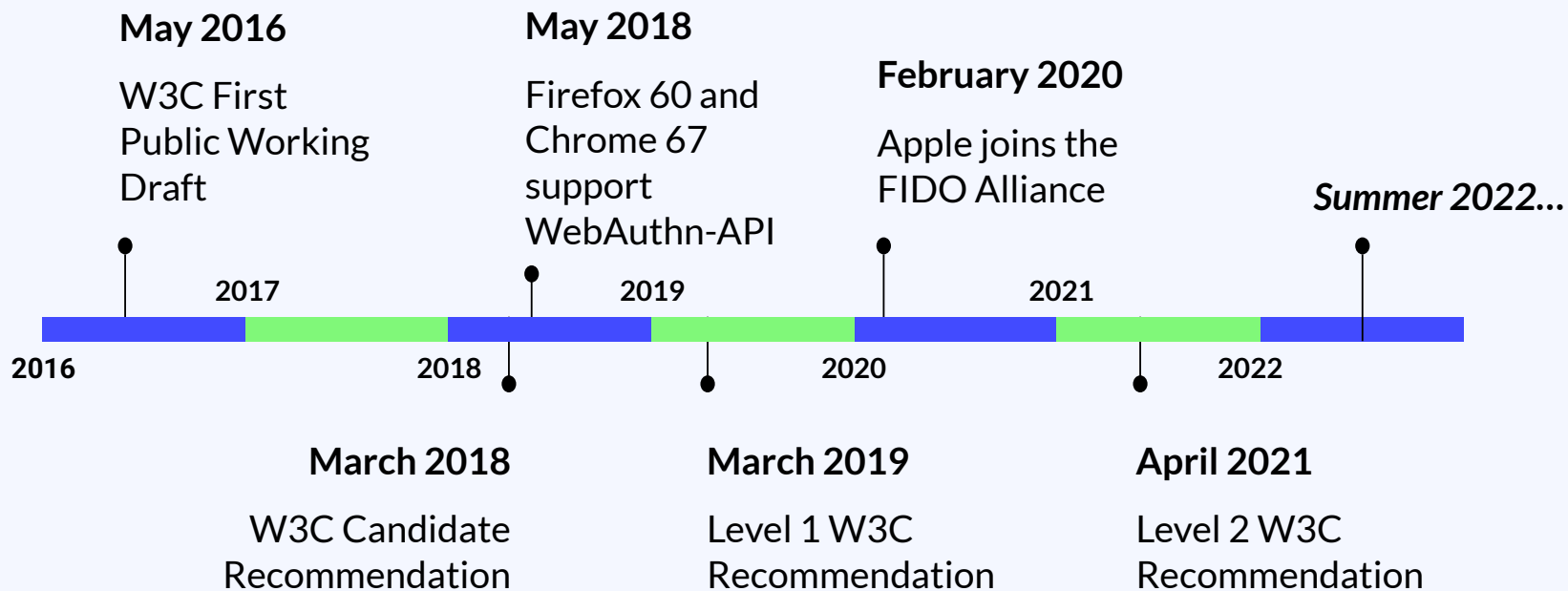
Registration ceremony



Authentication ceremony



Timeline of WebAuthn



Can I use WebAuthn?

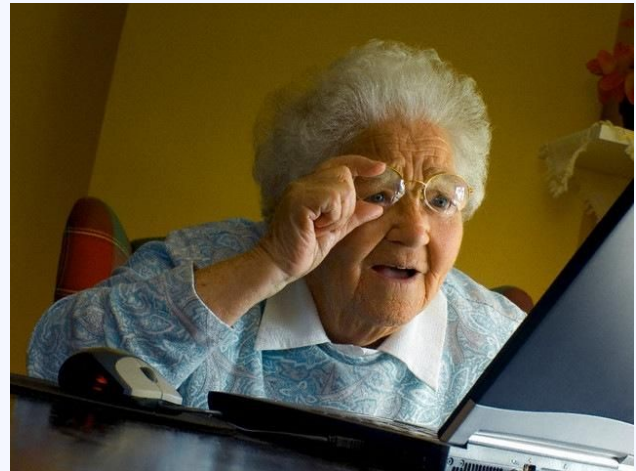
- today, >95% global usage possibility

Chrome	Edge *	Safari	Firefox	Opera	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	UC Browser for Android
						15.6			
						16.0			
						16.1			
						16.2			
						16.3			
						16.4			
109		15.6							
111	111	16.3	^{4 6} 111						
112	112	16.4	^{4 6} 112	97					
113	113	16.5	^{4 6} 113	98	113	16.5	20	all	13.4
114		16.6	^{4 6} 114						
115		TP	^{4 6} 115						
116									

caniuse.com, 22 May 2023

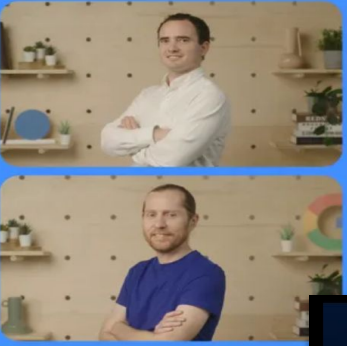
Usability problems with WebAuthn

- Passwords are known and widely understood
- Public key crypto is not
- Usage of external authenticators is new for most users
- Platform authenticators lack portability



⇒ Strong security requirements hinder the spread of WebAuthn

Android solutions
for seamless
sign-in across
devices



I/O

Google I/O 2022

Apple WWDC22

WWDC22

Meet passkeys

Garrett Davidson, Authentication Experience

Passkeys to the rescue

Passkeys vs WebAuthn

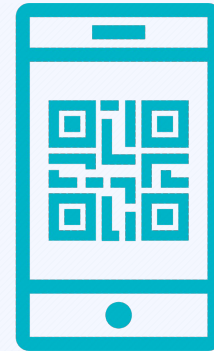
Passkeys are WebAuthn credentials + *usability*



Integrated



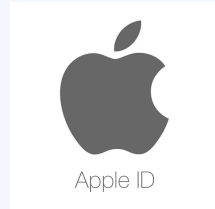
Synced



Portable

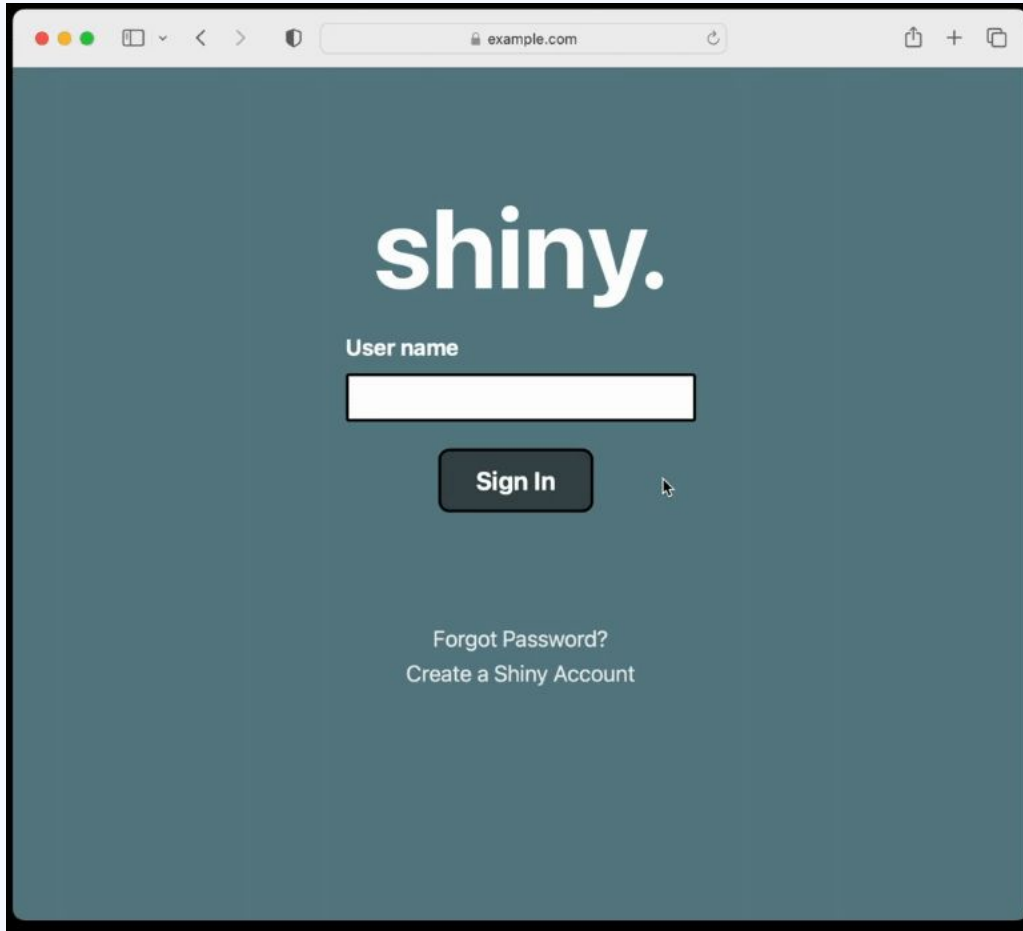
Improved integration into clients

- added to ecosystems and combined with user accounts



- using existent factors (Device lock / FaceID / TouchID)





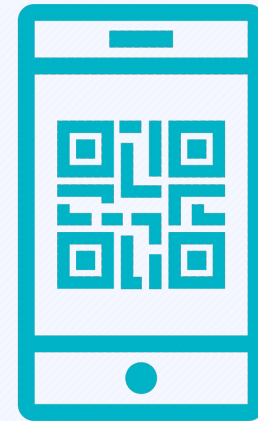
Synced

- Google:
 - Google Password Manager
 - e2e-encrypted
- Apple:
 - iCloud Keychain
 - e2e-encrypted
- Microsoft:
 - finally e2e-encrypted



Cross-device usage of passkeys

- own protocol for key exchange
Cross-Device Authentication
 - often powered by QR code
 - using CTAP
- security features
 - proximity check
 - e2e encrypted





Sign in



Sign in with passkey

or

Account name

Password

[Forgot password?](#)

Sign in

[Create an account](#)

5:01

Sat, Apr 9

72°F



Play Store



Gmail



Photos



TriBank



Using Passkeys like it's 2025

- increasing number of real passwordless applications



...

I want this! Passwordless in 5 steps

1. Extend your user entities
2. Plan enrollments of existing users
 - a. Detect support of Passkeys
 - b. Let users create credentials
 - c. Remove passwords
3. Allow registration of new passwordless users
4. Build passwordless login
5. Implement management of credentials



Integrating Passkeys

Implementing yourself

LIB

- using libraries
- available for all languages and frameworks

Passkeys as a service

API

- through API of Passkey provider
- flexibel own frontend integration

WCO

- using Web Components of Passkey provider
- out-of-the-box with theming possibilities

Comparing integration options

	LIB (Library)	API (Application Programming Interface)	WCO (Web Components)
Functional Scope	High	PK ceremonies via libraries	PK ceremonies via Web Comp. +
Development Effort	PK ceremonies via libraries	PK ceremonies via Medium (~50% of LIB) functionalities	PK ceremonies via Web Comp. + PK-CRUD functionalities + PK-UI/UX
Flexibility	High	High to Medium	Low
Data Security	Self-managed	Managed by Service	Managed by Service
Service Dependencies	No	Yes	Yes
Usage Fees	None	Required \$	Required \$\$\$
Session Management	Self	Self	Self or Service



Is there a budget for ongoing usage fees?

NO

LIB

YES

Are service dependencies acceptable?

NO

LIB

YES

Is limited data security acceptable?

NO

LIB

YES

Available time and capacity for development?

Rather more

Rather less

Knowledge of PK processes and PK-UI/UX?

LIB

Rather more

Rather less to none

Is increased flexibility due to complex authentication structures necessary?

LIB

Rather yes

Rather no

API

Is increased flexibility due to complex authentication structures necessary?

API

Eher ja

Rather nein

WCO

I want this! Passwordless in 6 steps

1. Extend your user entities
2. Plan enrollments of existing users
 - a. Detect support of Passkeys
 - b. Let users create credentials
 - c. Remove passwords
3. Allow registration of new passwordless users
4. Build passwordless login
5. Implement management of credentials
6. **Guide your users**



3 controversial opinions about Passkeys

- 1. Passkeys are not for the technical 5% of users
- they are for the masses**
- 2. Implementing Passkeys in your application
should be prioritized over MFA**
- 3. Most problems with Passkeys are problems
with processes, not technology**

Vielen Dank!



[Blogpost](#)



 @ClemensHuebner

 clemens.huebner@inovex.de

 @clemens@infosec.exchange

 @inovexgmbh

 @inovexlife