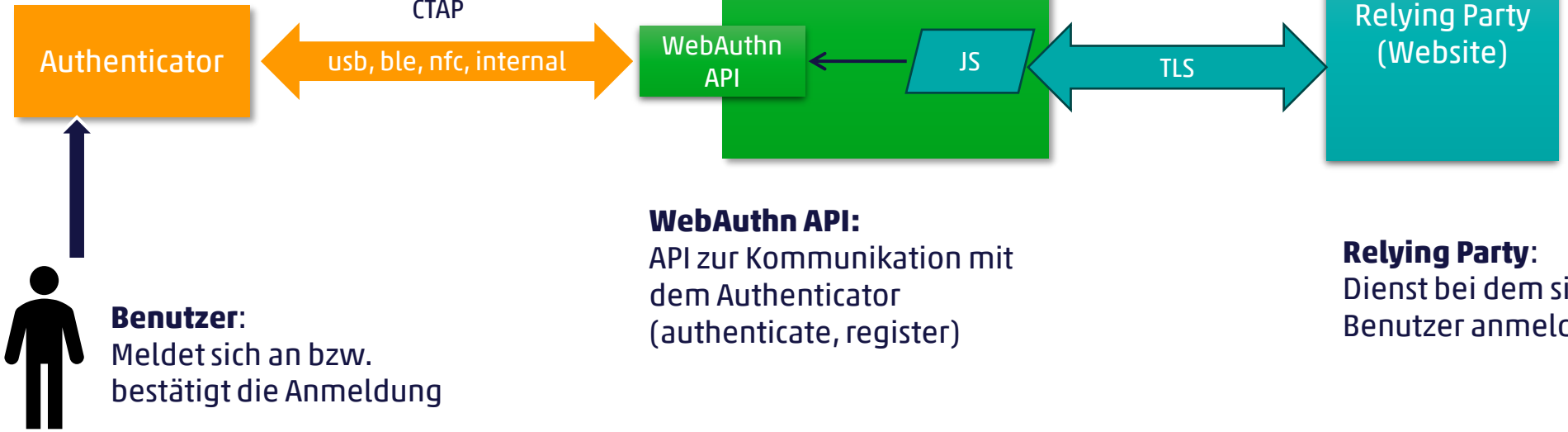
FIDO2 – Client Implementierung

Thomas Lüthi

cnlab Herbsttagung 2025 – FIDO2: einfach und sicher
Gleisarena, Zürich, 3. September 2025

cnlab

FIDO2 Authentisierung – Wer ist beteiligt?



RP JS im Client:
Startet die Registrierung bzw. die Anmeldung

WebAuthn API:
API zur Kommunikation mit dem Authenticator (authenticate, register)

Relying Party:
Dienst bei dem sich der Benutzer anmeldet

Benutzer:
Meldet sich an bzw. bestätigt die Anmeldung

Welche Authenticator gibt es?

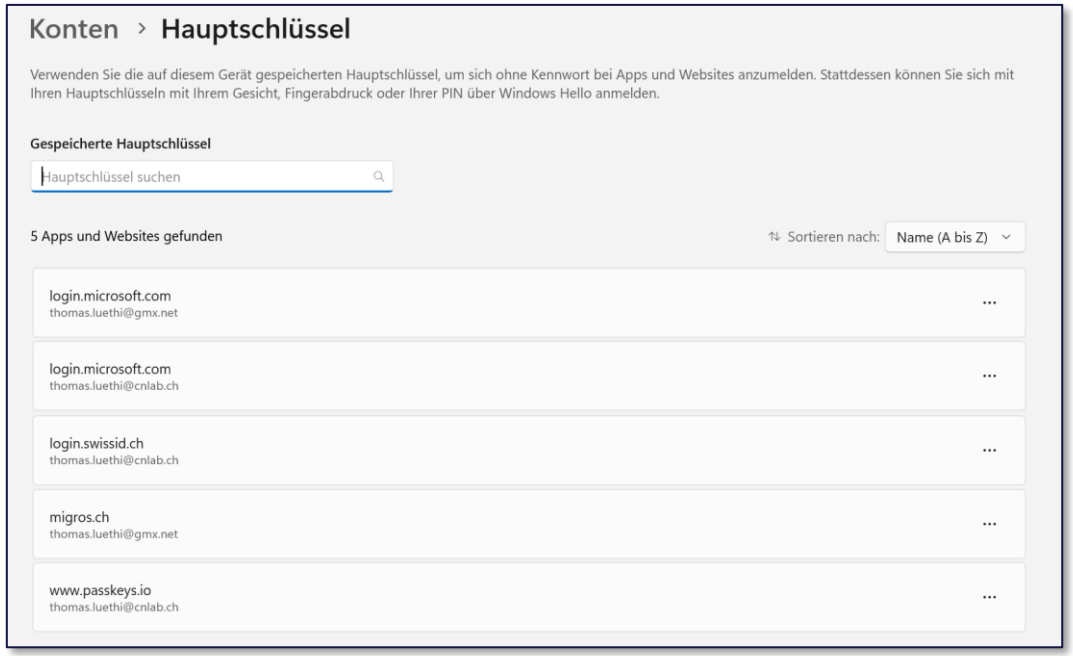
- Windows-Geräte (Windows Hello, TPM)
- Android-Geräte (Google Passwort Manager, Google Account)
- iOS-Geräte (iCloud Keychain, Apple-ID)

- Third-Party-Software (z.B. KeePassXC, LastPass, 1Password)

- YubiKey und andere Hardware-Tokens via USB/NFC

Windows Hello

- Verwaltung über die Windows 11 Einstellungen
- Die Schlüssel sind in der Windows Hello Credential-Datenbank verknüpft mit dem Benutzerkonto
- Privater Schlüssel: Gespeichert im TPM (Trusted Platform Module) oder in der Windows Hello Container-Plattform (bei Geräten ohne TPM)



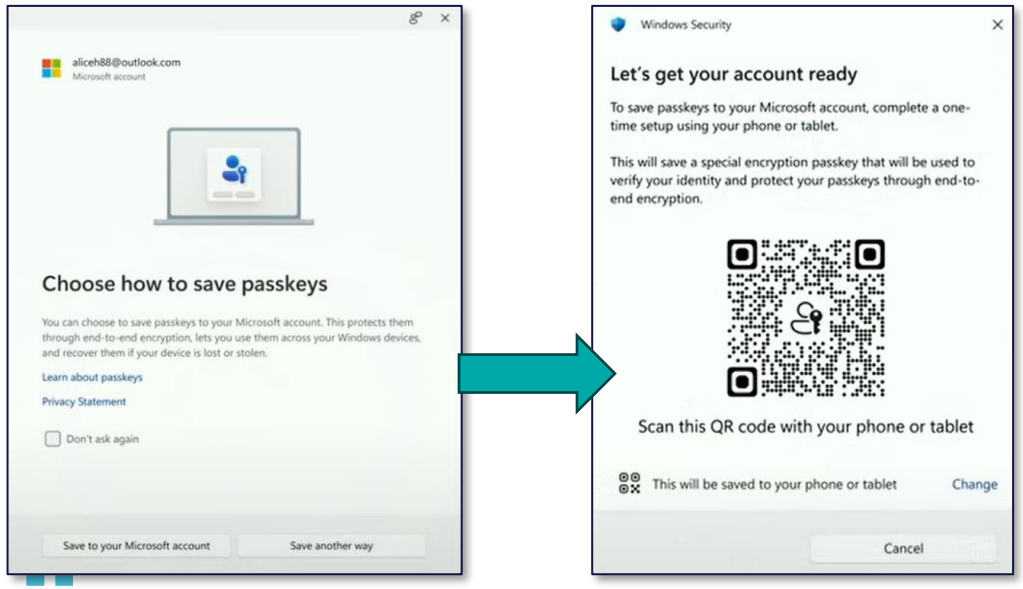
- Schlüsselattribut:

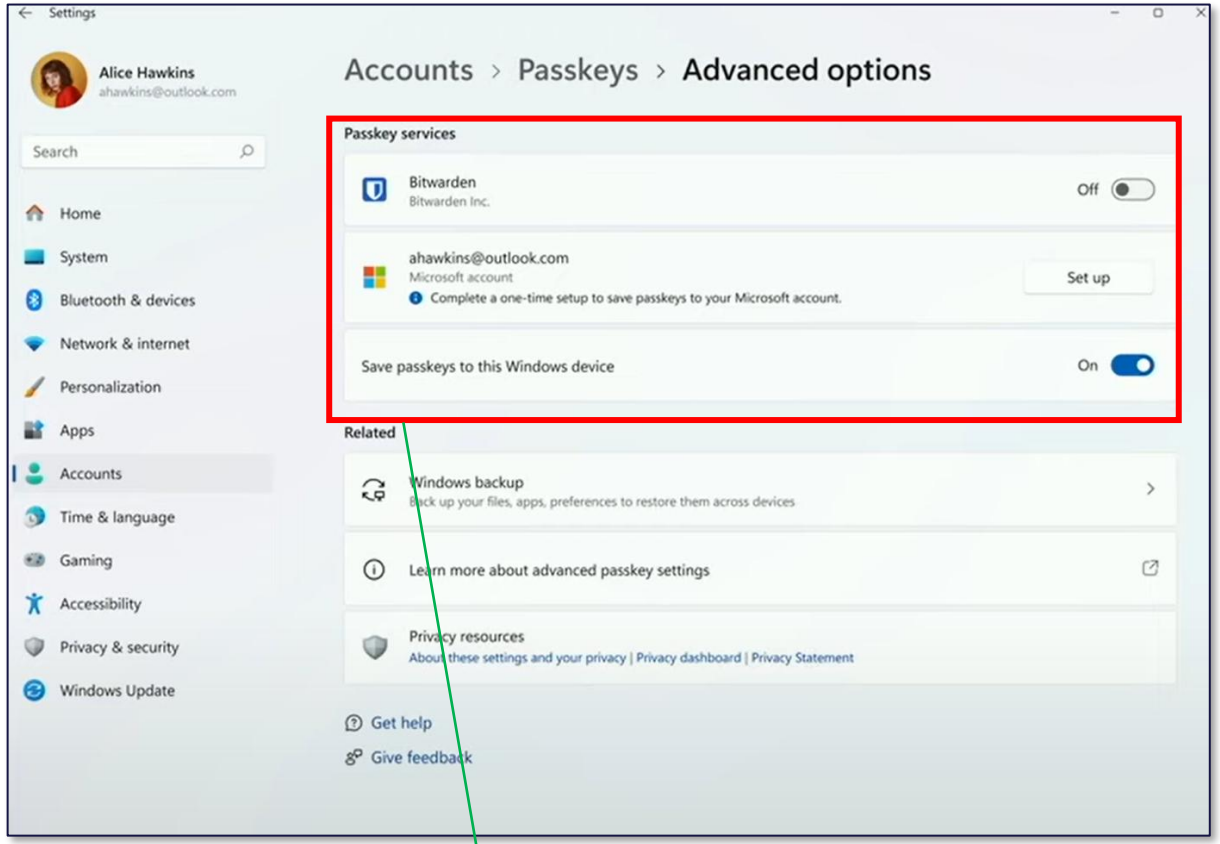
```
C:\certutil -csp "Microsoft Passport Key Storage Provider" -key -v
…
S-1-5-21-409369315-1216102017-581009308-1026/FIDO_AUTHENTICATOR/…
Schlüssel-ID-Hash(rfc-sha1): 4f18db2d4b02d2e61aaf2bbd5de58277367f0837
NgcKeyImplType: 1 (0x1)
    NCRYPT_IMPL_HARDWARE_FLAG - 1
…
S-1-5-21-3857130623-2685748287-314550012-1001/FIDO_AUTHENTICATOR/…
NgcKeyImplType: 2 (0x2)
    NCRYPT_IMPL_SOFTWARE_FLAG -- 2
…
```

Preview: Windows Passkey-Synch

- Verbesserter Passkey-Manger
- Synchronisation von Passkeys via Microsoft Cloud
- Aktuell erst im Insider-Preview-Build vorhanden

- Für die Synchronisation muss ein „Encryption“ Passkey erstellt werden:

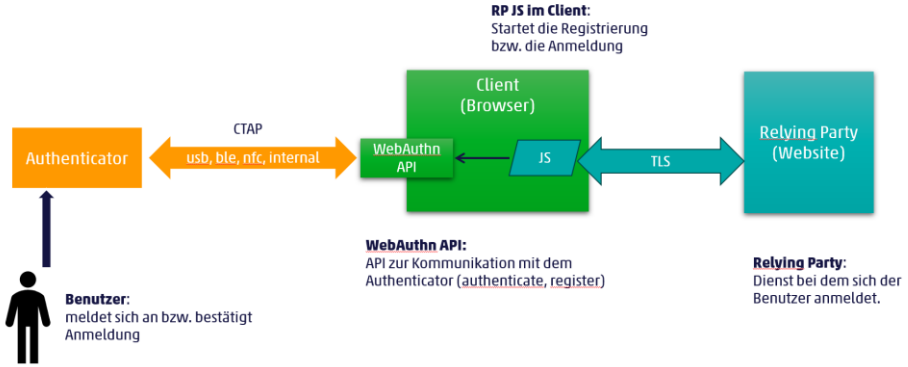




Plugin-Schnittstelle für Third-Party Passwort Providers vorhanden

Rolle des Browsers

- Implementiert das WebAuthn API für JavaScript
- Führt das JavaScript aus, das von der Relying Party kommt
- Kommuniziert mit dem Authenticator
- Phishing Protection



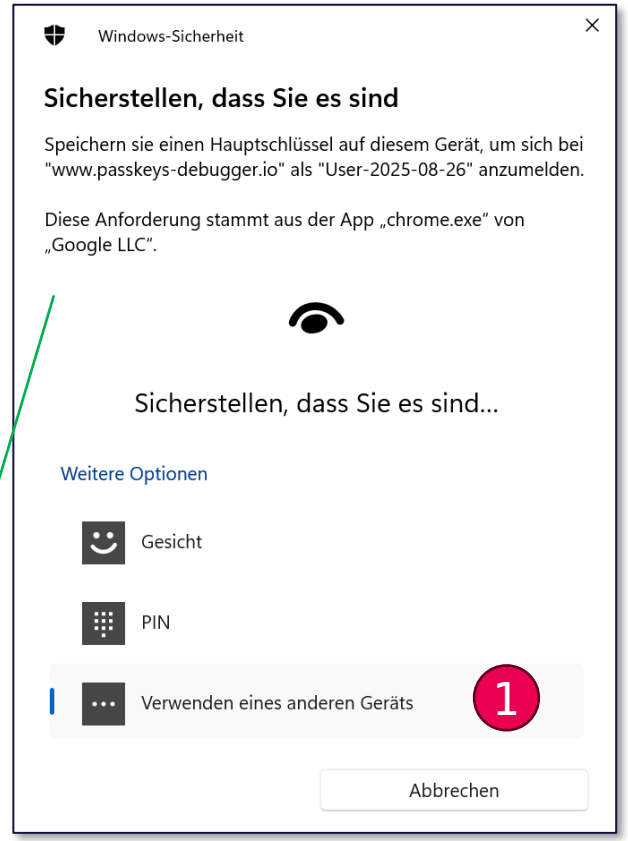
- Das JavaScript:
  - Verwendet das WebAuthn API um mit dem Authenticator zu kommunizieren (via CTAP)
    - Registrierung: `navigator.credentials.create()`
    - Authentisierung: `navigator.credentials.get()`
  - Muss allfällige Fehler vom WebAuthn API dem Benutzer kommunizieren
  - Muss das Resultat des API-Aufrufs der RP kommunizieren

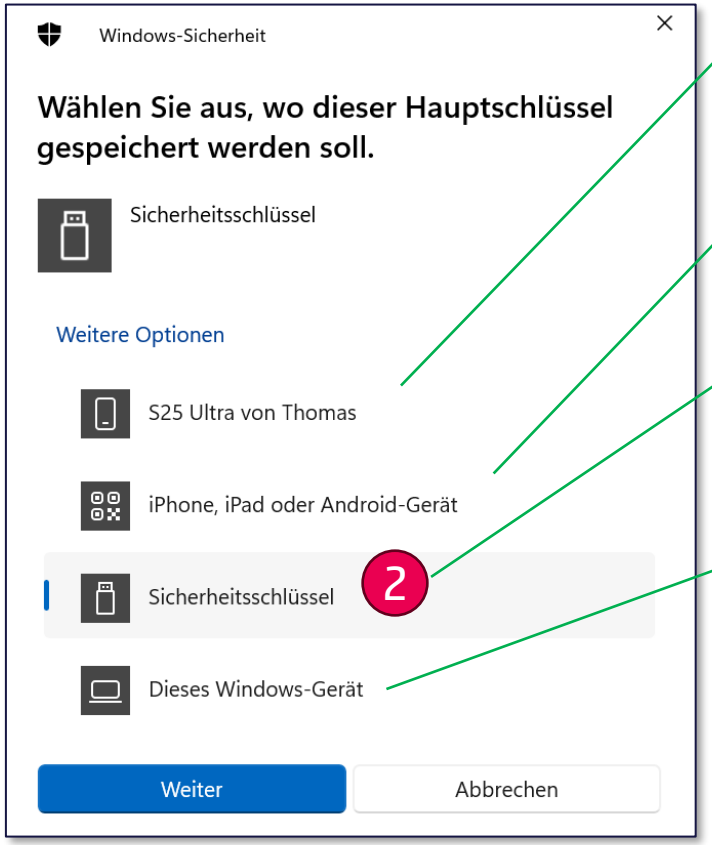
WebAuthn – Erzeugung eines Schlüssels (JavaScript)

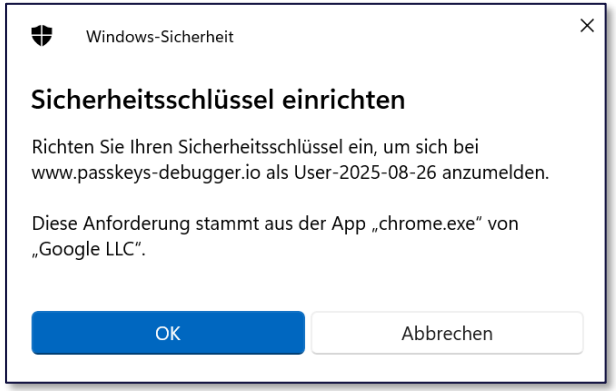
```
const publicKeyCredentialCreationOptions = {
 challenge,
 rp: {
      name,
      id},
 user: {
      id,
      name,
      displayName },
 pubKeyCredParams: {
      type,
      alg},
  authenticatorSelection: {
      residentKey,
      userVerification,
      authenticatorAttachment
 },
 attestation
};
```

```
navigator.credentials.create(
      publicKeyCredentialCreationOptions);
```

WebAuthn – Erzeugung eines Schlüssels (GUI) 1/2







Bereits „verbundenes" Smartphone

Smartphone via caBLE

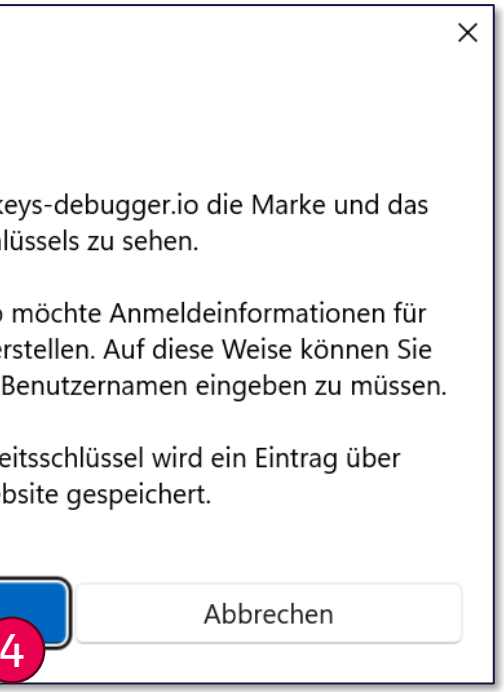
Hardware-Token

Zurück zu Windows Hello

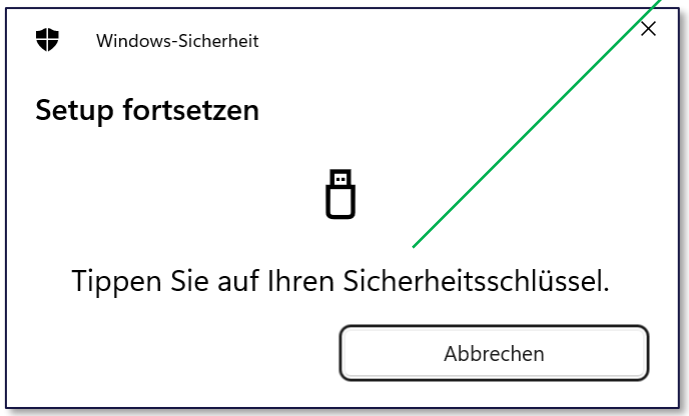
Windows Hello

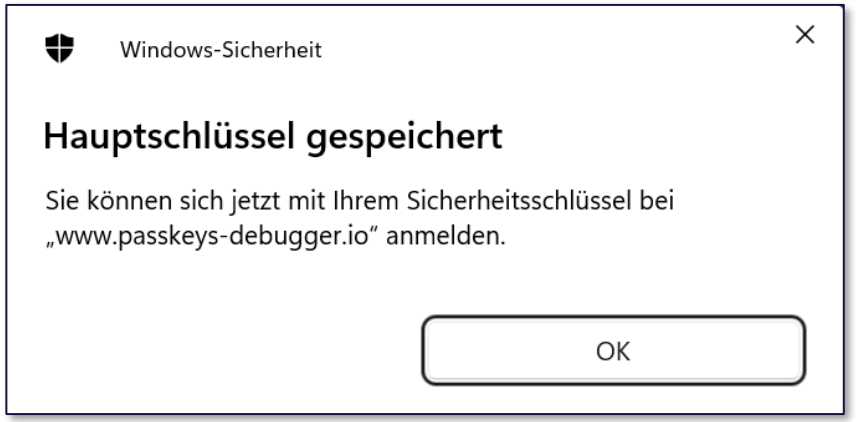
WebAuthn – Erzeugung eines Schlüssels (GUI) 2/2

User Präsenz bestätigen und Authentisierung mit Fingerprint



Sicherheitswarnung wegen der Attestation





WebAuthn – Antwort der Schlüsselerzeugung

- Die Registration Response wird bei der Erzeugung eines Schlüssels zurückgegeben und enthält Informationen über den Authenticator und den erzeugten Key
- Sie kann eine Signatur über einen Teil der Daten enthalten (Attestation Statement)
- Der Typen des Attestation Statement sagt mit welchem Key die Signatur erstellt wird:
  - Basic: Attestation Key ist spezifisch für ein Authenticator „Model“
  - Self: Der Credential Private Key wird verwendet
  - AttCA: Basierend auf dem TPM „Endorsement Key“ (EK) werden Authenticator spezifische Attestation-Keys bei der Attestation CA ausgestellt
  - AnonCA: Der Authenticator verwendet eine „Anonymization CA“ um dynamisch Credential-Keys zu erzeugen die nicht getrackt werden können
- In der Antwort muss auch die Zertifikatskette enthalten sein, um die Gültigkeit des Zertifikats zu prüfen

Es ist Aufgabe der Relying Party die Signatur und den verwendeten Authenticator zu prüfen!

```
{
  "authenticatorAttachment": "platform",
  "id": "34wlGez_UFqCfSb2Xpli-UQ4-S_V1zM-EoUempHwojOyGNvHQ-WFtdEv2h6kKOA6",
  "rawId": "34wlGez_UFqCfSb2Xpli-UQ4-S_V1zM-EoUempHwojOyGNvHQ-WFtdEv2h6kKOA6",
  "response": {
    "attestationObject": {
      "fmt": "packed",
      "attStmt": {
        "alg": "ES256 (-7)",
        "sig": "MEUCIHO-U5aMIh0fxOGTrxLrx60ccqWfVrUNbY2ln3qY8aBnAiEA2aGOrLrjL5xQAbHeo6SoxvBa7lSS3uIjyL-J3VTrQt0",
        "x5c": [
      },
      "authData": {
        "rpIdHash": "PpZrl-Wqt-OFfBpyy2SraN1m7LT0GZORwGA7-6ujYkM",
        "flags": {
          "userPresent": true,
          "userVerified": true,
          "backupEligible": false,
          "backupStatus": false,
          "attestedData": true,
          "extensionData": false
        },
        "counter": 1,
        "aaguid": {
          "raw": "2fc0579f-8113-47ea-b116-bb5a8db9202a",
          "name": "Unknown"
        },
        "credentialID": "34wlGez_UFqCfSb2Xpli-UQ4-S_V1zM-EoUempHwojOyGNvHQ-WFtdEv2h6kKOA6",
        "credentialPublicKey": "pAEBAycgBiFYIN-MJRns_1Bagn0m9l7qAXijlAwGVmpA4K2udJQHbaeC",
        "parsedCredentialPublicKey": {
          "keyType": "OKP (1)",
          "algorithm": "EdDSA (-8)",
          "curve": 6,
          "x": "34wlGez_UFqCfSb2XuoBeKOUDAZWakDgra50lAdtp4I"
        }
      },
      "clientDataJSON": {
        "type": "webauthn.create",
        "challenge": "8tRGqvo4bD_DBYDPoKEf0jvL3Yeh8-dRNycjZuiDx_U",
        "origin": "https://www.passkeys-debugger.io"
      },
      "transports": [
        "nfc",
        "usb"
      ],
      "authenticatorData": "PpZrl-Wqt-OFfBpyy2SraN1m7LT0GZORwGA7-6ujYkNFAAAAAS_AV5-BE0fqsRa7Wo25ICoAMN-MJRns_1Bagn0m9
      "publicKeyAlgorithm": -8
    },
    "type": "public-key",
    "clientExtensionResults": {}
}
```


Signatur

Daten die signiert werden

Es ist Aufgabe der Relying Party die Signatur und die durchgeführte Registrierung zur prüfen!

Authenticator	Client Plattform	Authenticator Verbindung	Authenticator Typ	Signatur
YubiKey	Windows	USB	cross-platform	yes
YubiKey	iOS	USB/NFC	platform	yes
Windows Hello	Windows	intern	platform	yes
Samsung Wallet	Android	intern	platform	yes
Samsung Wallet	Windows	BLE	cross-platform	none
iCloud Keychain	Windows	BLE	cross-platform	none
iCloud Keychain	iOS	intern	platform	none
Google Password Mgr	Windows	intern	platform	none
Google Password Mgr	Windows	BLE	cross-platform	none
Google Password Mgr	Android	intern	platform	none
KeypassXC	Windows	intern	platform	none

Nicht zuverlässig

Nicht zuverlässig

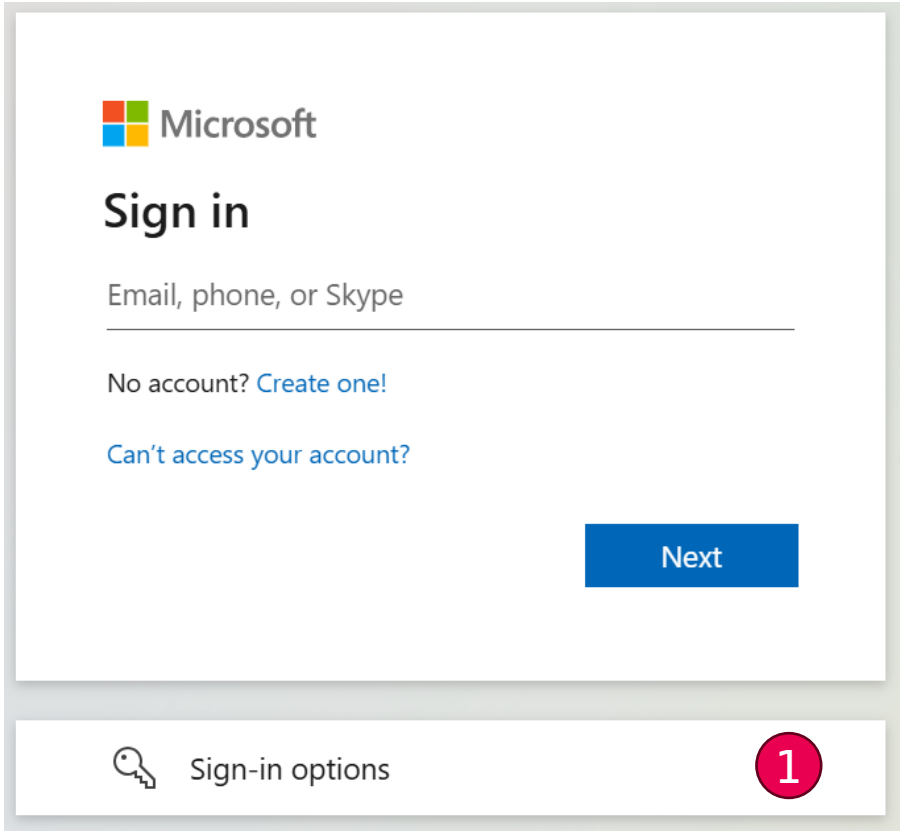
Keine Signatur bei Software Authenticator

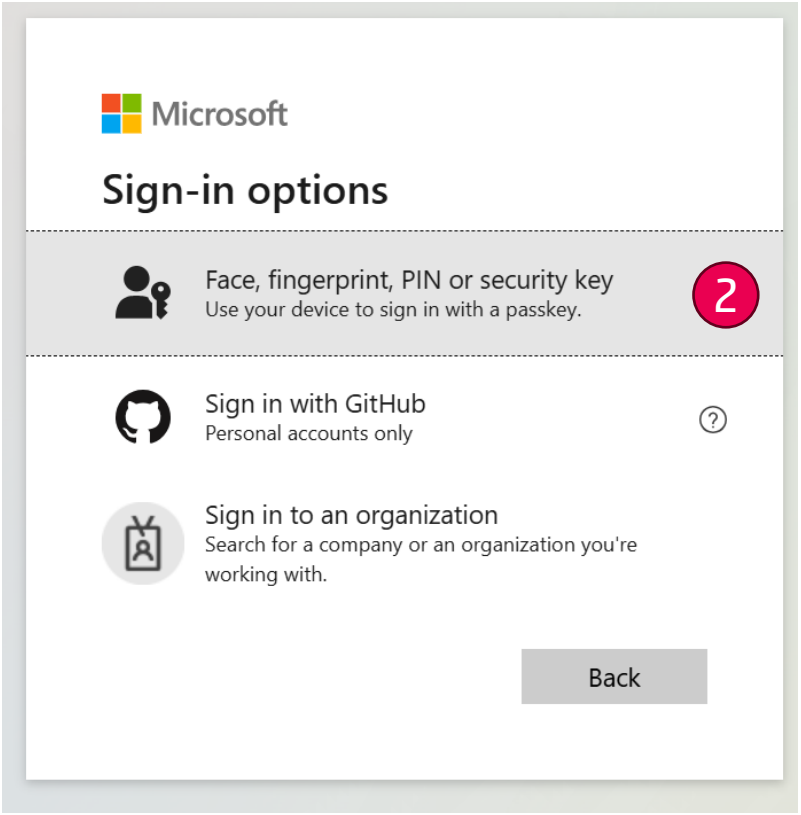
WebAuthn – Authentisierung (JavaScript)

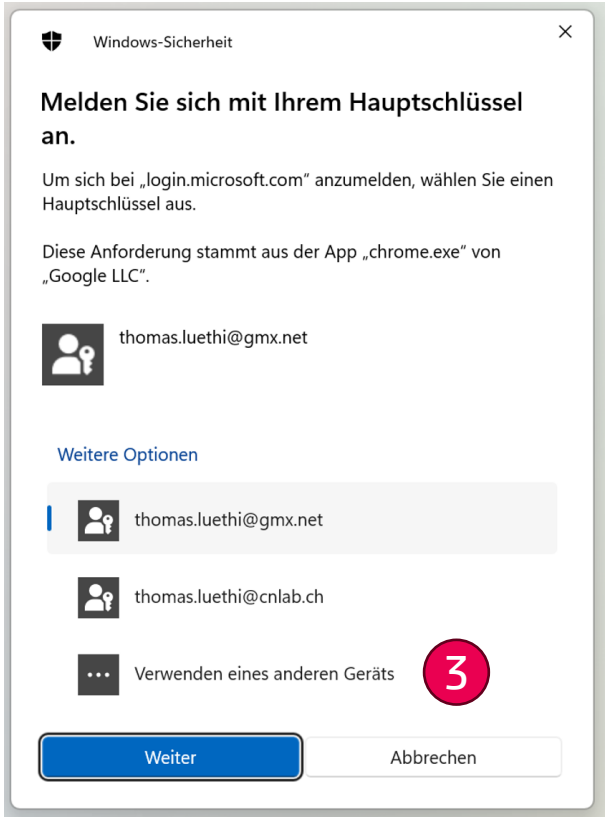
```
const publicKeyCredentialRequestOptions = {
  challenge,
  rpId,
  userVerification,
  allowCredentials
  };
```

```
navigator.credentials.get(
        publicKeyCredentialRequestOptions);
```

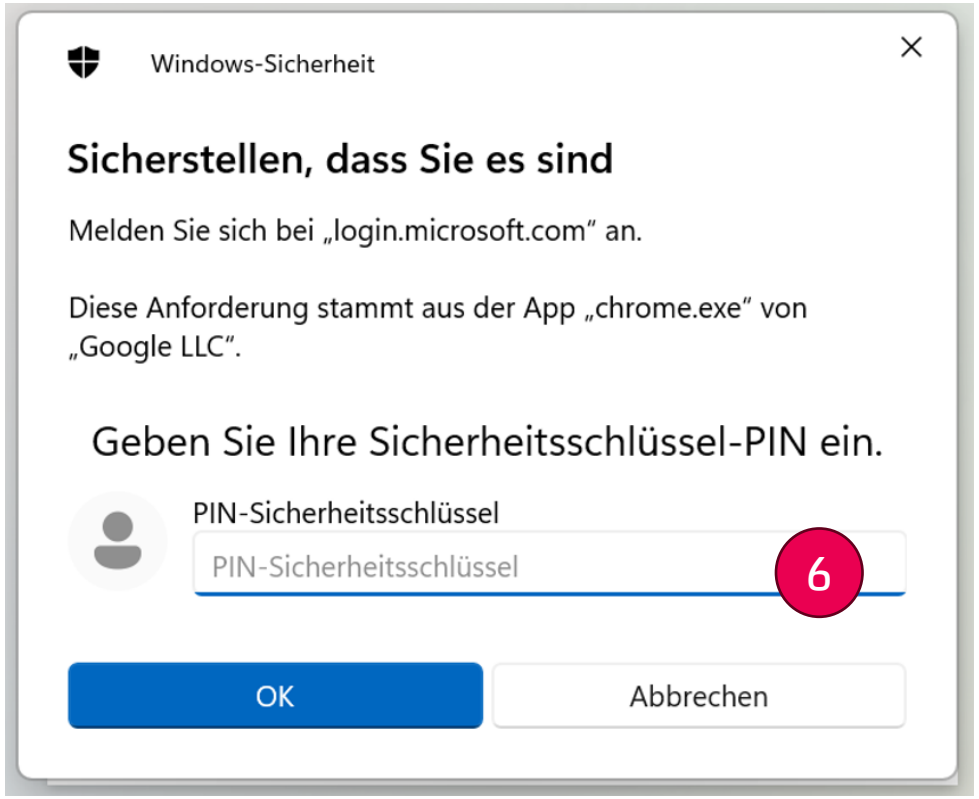
Authentisierung bei Microsoft Online mit FIDO-Token (GUI) 1/3

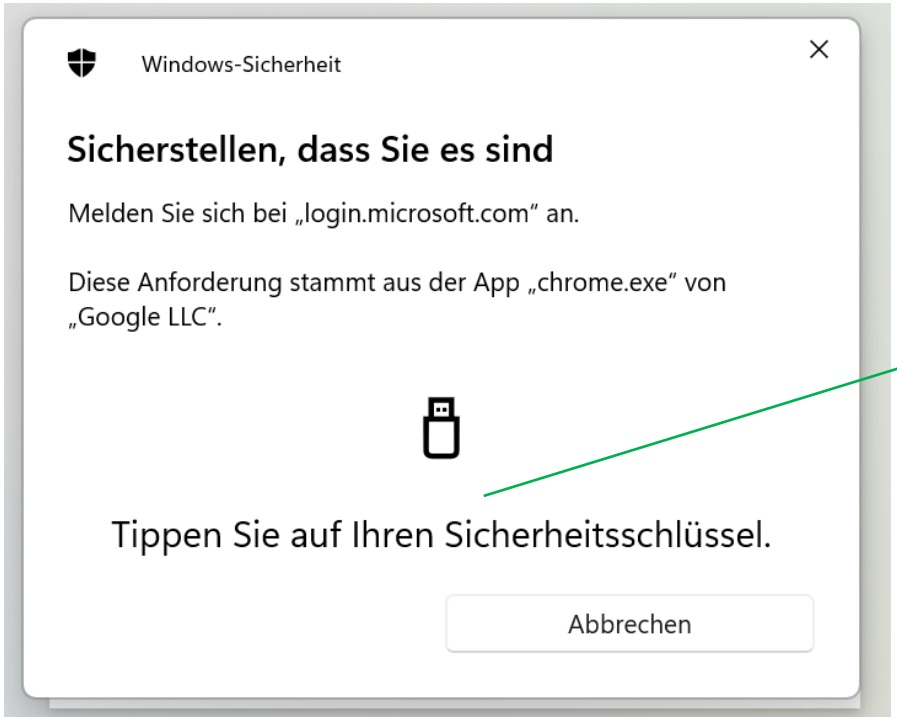






Authentisierung bei Microsoft Online mit FIDO-Token (GUI) 2/3

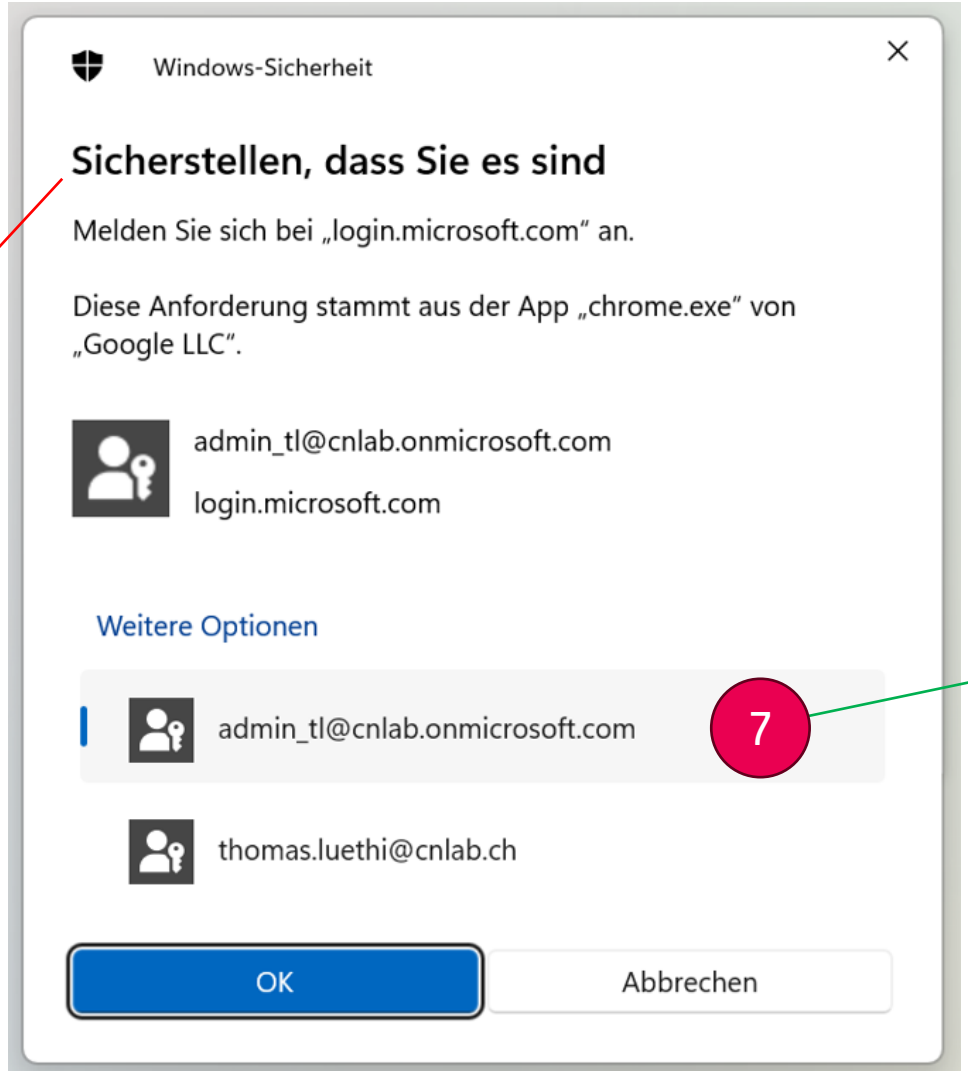




User Präsenz muss bestätigt werden

Authentisierung bei Microsoft Online mit FIDO-Token (GUI) 3/3

Text ist falsch, User Präsenz wurde bereits bestätigt im Schritt vorher geprüft



Auswahl des Passkey bei mehreren Passkeys auf dem Token.

WebAuthn – Antwort der Authentisierung


```
{
  "id": "zjrmesDCQroZsPQRsPJ9zZ3awEYGG6B4zmPyMqhTCzQ",
  "rawId": "zjrmesDCQroZsPQRsPJ9zZ3awEYGG6B4zmPyMqhTCzQ",
  "type": "public-key",
  "authenticatorAttachment": "platform",
  "response": {
    "authenticatorData": {
      "rpIdHash": "PpZrl-Wqt-OFfBpyy2SraN1m7LT0GZORwGA7-6ujYkM",
      "flags": {
        "userPresent": true,
        "userVerified": true,
        "backupEligible": false,
        "backupStatus": false,
        "attestedData": false,
        "extensionData": false
      },
      "counter": 1
    },
    "clientDataJSON": {
      "type": "webauthn.get",
      "challenge": "lviiYw8OB4J7PFc8YPtxrGPwlom6tUdRbfeZl2e2tac",
      "origin": "https://login.microsoft.com",
      "crossOrigin": false
    },
    "signature": "MEUCIC8c1obDeuF56RKDs-71fJSQCiVdDMlvyRpMoBc6WGfSAiEA6jfX8cvpuNboPwmPaC_iKAEzYPwokxH842PxPp0my4M",
    "userHandle": "Vt1yGcDGUPk_At_anEYyTYLNKBho-H1Zz_fCtr2w4ig"
  }
}
```


Informationen über den Passkey, den Authenticator und die Relying Party

Immer mit Signatur

Es ist Aufgabe der Relying Party die Signatur und die durchgeführte Authentisierung zur prüfen!

Plattformen mit FIDO2-Unterstützung ohne Browser

- Android
  - Android 9 (API 28) oder höher bietet das FIDO2 API
  - Android Credential Manager (ab Android 14)
- iOS
  - Apples Authentication Services Framework (seit iOS 16) unterstützt WebAuthn/FIDO2-Flows
- Windows
  - Microsoft Authentication Library (MSAL) für den Zugriff auf den Windows Authentication Manager (WAM) in .NET desktop applications
  - WebView2 unterstützt FIDO2

Vielen Dank für Ihre Aufmerksamkeit_

Thomas Lüthi
thomas.luethi@cnlab.ch
+41 55 214 33 41

info@cnlab-security.ch
+41 55 214 33 40

cnlab security AG
Obere Bahnhofstrasse 32b
CH-8640 Rapperswil-Jona
Switzerland

cnlab