Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Departement für Verteidigung, Bevölkerungsschutz und Sport VBS

Bundesamt für Cybersicherheit BACS

September 2025

Technologiebetrachtung

Passkeys

1 Einführung

Obwohl Passwörter viele bekannte und auch hinreichend gut dokumentierte sicherheitstechnische Nachteile haben, sind sie als Authentifikationsverfahren immer noch sehr populär und werden entsprechend verbreitet eingesetzt. Von den Authentifikationsverfahren, die in der Vergangenheit als Alternativen zu Passwörtern vorgeschlagen worden sind, hat sich bis heute keines durchgesetzt. Für den FIDO2-Standard und den darauf basierenden Passkeys sieht es diesbezüglich besser aus. In diesem Dokument werden der FIDO2-Standard und Passkeys kurz vorgestellt und im Hinblick auf ihre sicherheitstechnischen Eigenschaften diskutiert und eingeschätzt.

2 FIDO2-Standard

Die Fast IDentity Online (FIDO) Allianz[1] ist ein Zusammenschluss von industriellen Partnern mit dem gemeinsamen Ziel, passwortlose Authentifikationsverfahren als Alternativen zu Passwörtern zu entwickeln und zu standardisieren. Auf der Grundlage der früher verabschiedeten FIDO Universal Second Factor (U2F) und FIDO Universal Authentication Framework (UAF) Standards hat die FIDO-Allianz ein Client to Authenticator Protocol in der Version 2 (CTAP2) entwickelt, das zusammen mit dem Web Authentication (WebAuthn) Protokoll des World Wide Web Consortium[2] (W3C) den eigentlichen Kern des FIDO2-Standards darstellt.

Die Funktionsweise einer Authentifikation gemäss dem FIDO2-Standard und das Zusammenspiel der WebAuthn- und CTAP2-Protokolle sind in Abbildung 1 schematisch dargestellt. Grundsätzlich geht es darum, dass sich eine Benutzerin oder ein Benutzer über einen beliebigen Client gegenüber einem Server authentifizieren kann. Anstelle eines Passwortes wird eine dedizierte Hardware-Komponente («Authenticator») eingesetzt, die entweder im Client intern verbaut ist oder als externes Gerät (meist in der Form eines Tokens) an den Client angeschlossen wird. Dieser Anschluss kann kabelgebunden über eine Universal Serial Bus (USB) Schnittstelle oder kabellos über Bluetooth bzw. Near Field Communication (NFC) erfolgen. In allen Fällen findet die Authentifikation der Benutzerin oder des Benutzers zwischen

[1] https://fidoalliance.org
[2] https://www.w3.org

dem Authenticator und dem Server statt, und der Client verhält sich passiv (d. h. der Client vermittelt nur die Nachrichten, die zwischen dem Authenticator und dem Server ausgetauscht werden müssen). Das in Abbildung 1 blau hinterlegte WebAuthn-Protokoll definiert den diesbezüglichen Nachrichtenaustausch zwischen dem Client und dem Server, während das grün hinterlegte CTAP2-Protokoll den Nachrichtenaustausch zwischen dem Client und dem Authenticator beschreibt.

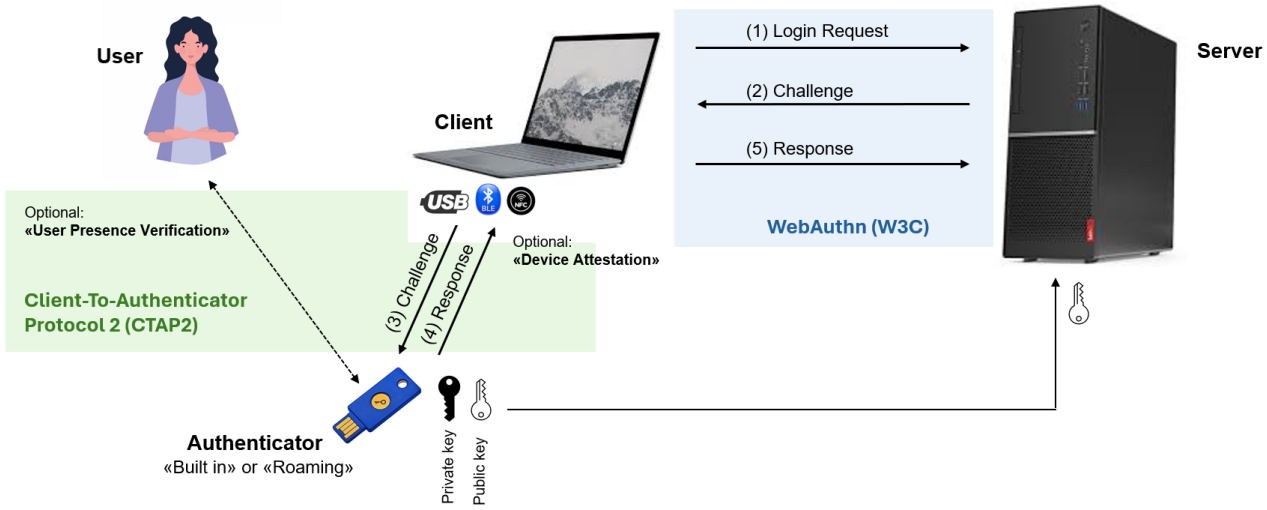

Abbildung 1: Authentifikation gemäss FIDO2-Standard

Damit ein Authenticator zur Authentifizierung einer Benutzerin oder eines Benutzers gegenüber einem über eine URL referenzierten Server eingesetzt werden kann, muss dieser zunächst einmal initialisiert werden. Dazu wählt der Authenticator für die Benutzerin oder den Benutzer ein zufälliges asymmetrisches Schlüsselpaar aus, das mit der URL des Servers assoziiert wird. Das Schlüsselpaar besteht aus einem privaten Schlüssel, mit dem digitale Signaturen erzeugt werden können, und einem öffentlichen Schlüssel, mit dem digitale Signaturen verifiziert werden können. Der private Schlüssel bleibt auf dem Authenticator gespeichert, während der öffentliche Schlüssel dem Server bekanntgegeben wird.

Nach dieser Initialisierung kann sich die Benutzerin oder der Benutzer mit dem Authenticator gegenüber dem Server mit Hilfe eines einfachen Challenge-Response-Verfahrens bzw. -Protokolls authentifizieren. Dieses Protokoll besteht im Wesentlichen aus fünf Schritten: In Schritt (1) gelangt der Client mit einem Login Request an den Server. Der Server stellt dem Client in Schritt (2) eine Challenge zu, die mit dem privaten Schlüssel signiert werden muss, um eine gültige Response zu erhalten. Diese Signatur kann nur vom Authenticator mit dem privaten Schlüssel erzeugt werden. In Schritt (3) stellt der Client dem Authenticator dazu die Challenge zu und in Schritt (4) antwortet dieser mit einer passenden Signatur als Response. Schliesslich übergibt der Client diese Response dem Server in Schritt (5). Mit Hilfe des öffentlichen Schlüssels kann der Server dann die Gültigkeit der Response und damit auch die Authentizität der Benutzerin oder des Benutzers überprüfen.

Weil die URL des Servers in die Challenge mit eingebunden ist und vom Authenticator vor der Erzeugung der Signatur überprüft wird, kann die Response nur zur Authentifikation gegenüber dem eigentlichen Zielserver eingesetzt werden. Damit kann das WebAuthn-Protokoll die Benutzerin oder den Benutzer wirksam vor vielen Phishing- und «Mallory-in-the-Middle» (MITM)

Angriffen schützen.[3] Während eines Protokolllaufes können zudem der Client die Zulässigkeit des Authenticators («Device Attestation») und der Authenticator die Präsenz und Einwilligung der Benutzerin oder des Benutzers überprüfen («User Presence Verification»). Diese Überprüfungen sind aber optional und können von Implementierung zu Implementierung variieren. Im einfachsten und häufigsten Fall wird über die Device Attestation der Einsatz eines bestimmten Tokens erzwungen und die Präsenz und Einwilligung der Benutzerin oder des Benutzers wird dadurch überprüft, dass eine bestimmte Taste (auf dem Token oder Client) betätigt werden muss.

3 Passkeys

Ursprünglich ist es beim FIDO2-Standard darum gegangen, den privaten Schlüssel einer Benutzerin oder eines Benutzers für eine bestimmte Applikation sicher und nicht auslesbar im Authenticator zu speichern. Damit ist die Authentizität der Benutzerin oder des Benutzers fest an diesen Authenticator gebunden («device-bound authenticators»). Das ist aus sicherheitstechnischer Sicht zwar wünschenswert, aus praktischer Sicht ergeben sich aber Schwierigkeiten beim Einsatz mehrerer auch unterschiedlicher Authenticators - vor allem wenn sie intern und in Clients verbaut sind. Einfacher wäre es, wenn die Authenticators einer Benutzerin oder eines Benutzers die von ihnen gespeicherten und verwalteten privaten Schlüssel z. B. über einen Cloud-Speicher synchronisieren könnten («synced authenticators»). Damit liesse sich eine gewisse Geräteunabhängigkeit erreichen. Die entsprechende Technologie ist von der FIDO-Allianz ebenfalls erarbeitet worden und wird als «Passkeys» bezeichnet. In der Tat bezeichnet man heute als Passkeys sowohl diese Technologie als auch die privaten Schlüssel, die mit Hilfe dieser Technologie zur Authentifikation anstelle von Passwörtern eingesetzt werden können.

Für die Implementierung von Passkeys gibt es viele Freiheitsgrade und Vereinfachungsmöglichkeiten, die allerdings alle zu Lasten der Sicherheit gehen. Wie erwähnt können die Passkeys einer Benutzerin oder eines Benutzers geräteübergreifend synchronisiert werden. Einige Implementierungen unterstützen darüber hinaus auch das Credential Exchange Format (CXF) und das entsprechende Credential Exchange Protocol (CXP).[4] Damit können Passkeys auch einzeln exportiert und auf anderen Geräten wieder importiert werden, wobei dieser Austausch manchmal sogar benutzerübergreifend möglich ist. Für die Nutzung von sich physisch in der Nähe befindlichen Authenticators gibt es die Möglichkeit einer Cross Device Authentication (CDA), damit auch Geräte in das Passkeys-Ökosystem eingebunden werden können, die selbst Passkeys nicht unterstützen. Schliesslich sind auch im Bereich des ursprünglich vorgesehenen Hardware-Schutzes Vereinfachungen möglich, so dass z. B. Passwortmanager auch ohne dedizierten Hardware-Schutz Passkeys verwalten und die abgelegten Passkeys softwaremässig schützen können. Entsprechend wird man bei einer konkreten Implementierung von Passkeys sehr genau schauen müssen, wie stark im Hinblick auf eine bessere Benutzerfreundlichkeit vom ursprünglichen FIDO2-Standard abgewichen wird und was die entsprechenden sicherheitstechnischen Implikationen sind. Grundsätzlich sieht die FIDO-Allianz hier drei (Sicherheits-) Stufen für Authenticators bzw. deren Restricted Operating

[3] Bei einem MITM-Angriff tritt die oder der Angreifende zwischen dem Client und dem Server in Erscheinung, d. h. der Client und der Server kommunizieren je mit dem MITM, glauben aber, direkt miteinander zu kommunizieren. Als Alternative oder Ergänzung zu diesem Sicherheitsmechanismus, der auf die Einbindung der URL des Zielservers abzielt, kann auch das Token-Binding-Protokoll eingesetzt werden, das als Erweiterung von TLS in den RFC-Dokumenten 8471 und 8472 spezifiziert ist. Allerdings wird TLS-Token Binding von handelsüblichen Browsern nur schlecht unterstützt.

[4] https://fidoalliance.org/specs/cx/

Environment (ROE) vor,[5] wobei die Granularität dieser (drei) Stufen für effektive Sicherheitsbetrachtungen und -konzipierungen nicht ausreichend ist.

4 Einschätzung

Der FIDO2-Standard ist aus sicherheitstechnischer Sicht positiv zu werten. Anstelle von Passwörtern werden moderne kryptografische Verfahren eingesetzt, die die Benutzerin oder den Benutzer entlasten. Der ursprünglich angedachte Hardware-Schutz bzw. die Nichtauslesbarkeit des entsprechenden Schlüsselmaterials ist aber im Hinblick auf eine grosse Marktdurchdringung mit Passkeys aufgegeben worden. So bieten Passkeys heute eine sehr grosse Flexibilität, um möglichst viele Einsatzgebiete abdecken zu können.

Weil aus der Sicht der Sicherheit der Teufel immer im Detail steckt und diese Details auch von der jeweiligen Implementierung abhängen, können über die Sicherheit von Passkeys keine allgemein gültigen Aussagen gemacht werden. Aus mindestens drei Gründen sind aber Passkeys immer besser als Passwörter:

(1) Im Gegensatz zu Passwörtern sind Passkeys pro Zielserver unterschiedlich und müssen damit einzeln kompromittiert werden.[6]

(2) Im Gegensatz zu Passwörtern sind Passkeys pseudozufällig erzeugt und können entsprechend auch nicht einfach erraten werden.

(3) Im Gegensatz zu Passwörtern werden Passkeys nicht über Datenkommunikationsleitungen übertragen und können entsprechend auch nicht aus dem Datenverkehr herausgelesen und für «Replay»-Angriffe missbraucht werden.

Leider werden viele Einsatzgebiete von Passkeys aber die Weiternutzung von Passwörtern ermöglichen oder sogar erfordern (z. B. als Backup-Lösung für das Zurücksetzen von Passwörtern), so dass schwache Passwörter und vor allem auch schlechte Passwortverwaltungsprozesse immer noch realistische Angriffsmöglichkeiten darstellen. Diese Tatsache gilt es bei der Beurteilung von auf Passkeys aufsetzenden Sicherheitsdispositiven mitzuberücksichtigen.

Aufgrund seiner sicherheitstechnischen Vorteile und der breiten industriellen Unterstützung geht das Bundesamt für Cybersicherheit (BACS) davon aus, dass sich der FIDO2-Standard in der Ausprägung von verschiedenen Passkeys-Implementierungen auf dem Markt durchsetzen wird. Die grossen Technologieanbieter wie Google, Apple und Microsoft gehen jedenfalls in diese Richtung und ersetzen, wo immer möglich, Passwörter durch Passkeys. Damit werden die Voraussetzungen für eine passwortlose Authentifikation geschaffen. Das BACS begrüsst und unterstützt diese Entwicklung.

---

[5] Gemäss https://fidoalliance.org/certification/authenticator-certification-levels/ ist bei der ersten Stufe (L1) der Einsatz von beliebigen Hard- und Software-Komponenten möglich. Die zweite Stufe (L2) erfordert den Einsatz einer ROE in Form eines Secure Elements (SE) bzw. einer Trusted Execution Environment (TEE), und die dritte Stufe (L3) sogar den Einsatz einer von der FIDO-Allianz genehmigten Allowed ROEs (AROEs).

[6] Demgegenüber sind Passwörter so konzipiert und werden auch so eingesetzt, dass einzelne Passwörter für viele Server eingesetzt werden oder die einzelnen Passwörter sich nur geringfügig unterscheiden können.

Abkürzungen

AROE	Allowed ROE
BACS	Bundesamt für Cybersicherheit
CDA	Cross Device Authentication
CTAP	Client to Authenticator Protocol
CXF	Credential Exchange Format
CXP	Credential Exchange Protocol
FIDO	Fast IDentity Online
MITM	Mallory-in-the-Middle
NFC	Near Field Communication
RFC	Request for Comments
ROE	Restricted Operating Environment
SE	Secure Element
TEE	Trusted Execution Environment
TLS	Transport Layer Security
U2F	Universal Second Factor
UAF	Universal Authentication Framework
URL	Uniform Resource Locator
USB	Universal Serial Bus
W3C	World Wide Web Consortium
WebAuthn	Web Authentication