Das Passwort ist tot – es leben die Passkeys

Das Passwort ist tot – es leben die Passkeys

Passwörter sind vom Konzept her kaputt, da helfen auch starke Passwörter und klassische Mehr-Faktor-Authentifizierung (MFA) nicht. Stattdessen brauchen wir konzeptionell sichere Authentifizierungsverfahren wie Passkeys.

Von Jürgen Schmidt

-tract

- Die Authentifizierung mit Passwörtern ist inhärent unsicher: Auch die klassische Mehr-Faktor-Authentifizierung lässt sich mit Phishing aushebeln.
- Beim Passkey-Verfahren errechnet ein Authenticator für jede Domain einen eigenen Secret Key, mit dem er die Antwort auf eine Challenge des Servers signiert. Diese Antwort authentifiziert den Anwender.
- Passkey und Secret Key verlassen nie den Hoheitsbereich des Users.
- Der Authenticator kann ein externes Token oder im Betriebssystem integriert sein. Windows, macOS, Android und iOS bieten die Funktion bereits.

Das Grundproblem der Authentifizierung lösen Passwörter mit einem Geheimnis, dessen Besitz die Identität beweisen soll: Ich bin „ju“ und nur ich kenne das Geheimnis IM~qaaU0h!N5Z-:(UR~{H;8. Das ist noch nicht das Problem, sondern ein durchaus legitimes Konzept, mit dem sich prinzipiell bereits

eine recht hohe Sicherheitsstufe erreichen ließe. Das Problem beginnt an der Stelle, an der ich IM~qaaU0h!N5Z-:(UR~{H;8 einem Dienst als Antwort auf dessen Anfrage „Wie lautet dein Passwort?“ sende, um meine Identität zu beweisen.

Ich überspringe hier absichtlich die Klagen über 123456 und ficken123 als Passwörter, die sich viel zu einfach knacken oder erraten lassen. Denn das lässt sich durch entsprechende Policies, Awareness-Schulungen und den Einsatz von Passwort-Safes noch einigermaßen in den Griff bekommen. Doch auch mein Superpasswort, das ich nirgends anders benutze, ist eigentlich schon in dem Moment kompromittiert, in dem ich es in ein Passworteingabefeld eintippe respektive kopiere und abschicke.

Abgephisht

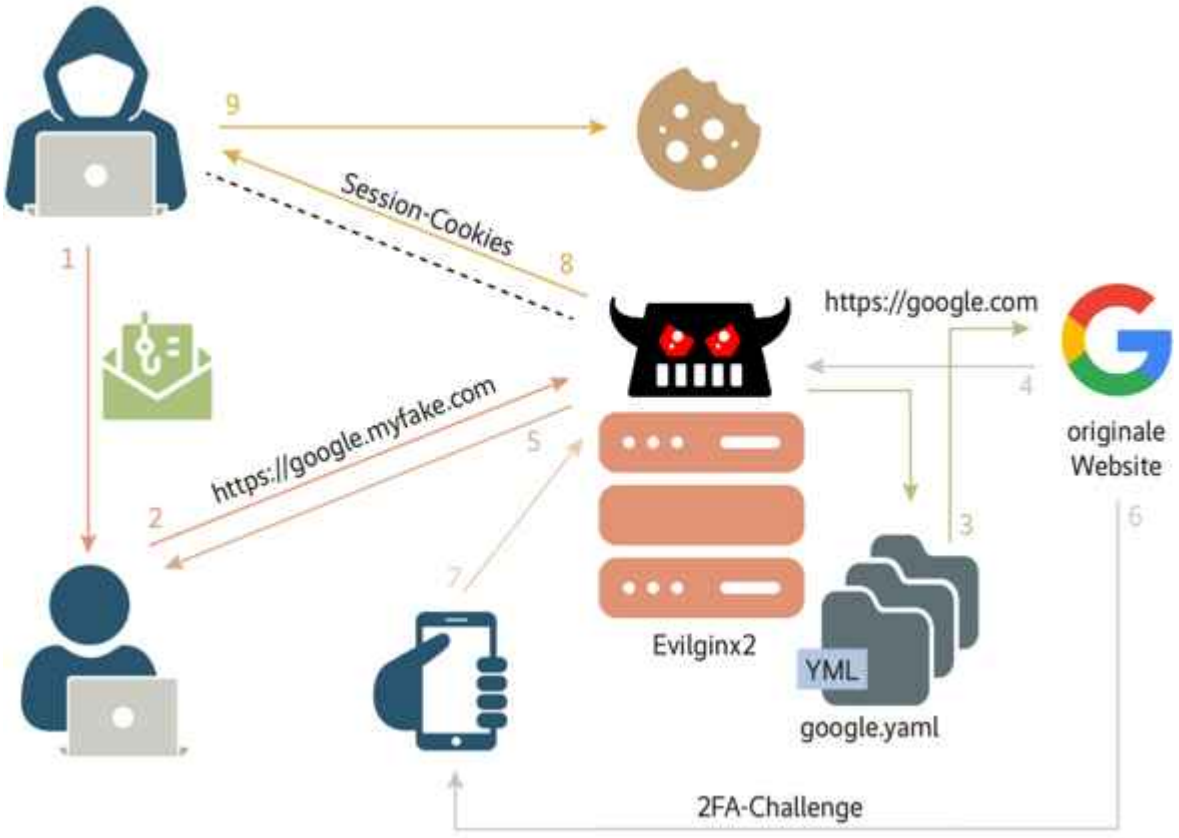
Denn damit ist das Geheimnis nicht mehr geheim. Letztlich weiß ich nicht, wer da auf der Empfängerseite mein geheimes Passwort bekommt und was der oder die dann damit macht. Phishing ist einer der wichtigsten Angriffsvektoren – und eine Gefahr sowohl für Privatpersonen als auch für Unternehmen. Das zeigen nicht nur Statistiken. Professionelle Penetrationstester erklären mir regelmäßig: „Wenn sonst gar nichts geht – Phishing geht immer.“ Und das gilt vermehrt auch für Zwei-Faktor-Authentifizierung.

Denn was bringt es, wenn ich zwei voneinander unabhängige Identitätsnachweise habe und dann letztlich zwei Zeichenketten an den Phisher schicke? Also beispielsweise mein normales Passwort und den Einmalcode, den ich via SMS oder von einer TOTP-App wie dem Authenticator bekommen habe? So gut wie nichts. Denn der Phisher muss nur schnell genug sein, sich während des Gültigkeitszeitraums mit den abgephishten Credentials beim Server anzumelden – und er ist drin.

Echtzeit-Phishing

Wie leicht das geht, demonstriert das gern für Phishingtests

eingesetzte Tool Evilginx2 eindrucksvoll. Es funktioniert ähnlich wie der beliebte HTTP-Cache und Reverse-Proxy nginx. Damit setzen Angreifer mit wenigen Handgriffen eine Phishingseite auf, die dem Original gleicht wie ein Ei dem anderen. Denn alle Inhalte stammen von der echten Seite. Und die gesamte Kommunikation reicht der böse Ginx in Echtzeit in beide Richtungen durch.



Evilginx2 setzt sich als Man in the Middle zwischen Anwender und Webseite und kann die Websession nach dem Anmelden trotz Zwei-Faktor-Authentifizierung übernehmen (Abb. 1).
Allerdings nicht ganz direkt – zunächst protokolliert er alle Passwörter und auch das nach einer erfolgreichen 2FA ausgetauschte Session-Cookie. Da braucht es je nach dem von der Website eingesetzten Verfahren zum Session-Management etwas Anpassung – aber danach schreibt Evilginx2 dieses wichtige Token mit. Und damit hat der Phisher vollen Zugriff auf den Dienst. Da helfen weder SMS-PIN, TOTP noch Push-Authentication. Solange der Phisher die übertragenen Daten einfach weiterreichen und sich damit erfolgreich am Server anmelden kann, ist er drin.

Somit muss der Angreifer es nur noch schaffen, sein Opfer auf diese Phishingseite zu locken. Und dazu hat er beliebig viele Versuche und kann systematisch alle Mitarbeiter eines Unternehmens durchprobieren. Immer wieder. Irgendwann klickt eine oder einer auf den Link in der auf sie oder ihn zugeschnittenen Mail oder in dem als Waterhole zusammengezimmerten Forum. Natürlich kann man fordern, dass Anwender die URL einer Website prüfen müssen, bevor sie ihre Zugangsdaten eintippen. Aber immer öfter sieht man die URL gar nicht mehr – etwa auf Mobilgeräten oder bei Pop-up-Fenstern. Da hilft letztlich auch kein Awareness-Training, denn schon ein einziges unaufmerksames Opfer genügt. Wie die Pentester sagen: „Phishing geht immer."

FIDO for the win

Um das zu ändern, hat die Alliance für Fast Identity Online (FIDO) ein konzeptionell besseres Authentifizierungsverfahren entworfen. Man hat dabei nach wie vor ein Geheimnis zum Nachweis der Identität – aber das gibt man nicht mehr Hinz und Kunz, sondern behält es immer unter seiner eigenen Kontrolle. FIDO ersetzt also das als Shared Secret fungierende Passwort, das man ständig durch die Gegend schickt, durch ein echtes Geheimnis – den Passkey.

Das ist keineswegs nur Wortklauberei, sondern ermöglicht den Einsatz moderner kryptografischer Verfahren, die viele Angriffe auf Passwörter praktisch unmöglich machen. Das Passkey-Konzept beruht auf asymmetrischer Kryptografie und einem Challenge-Response-Verfahren. Dazu benötigt man ein Stück Software, den sogenannten Authenticator, der den Authentifizierungsvorgang auf der Clientseite abwickelt. Er erzeugt für jeden Dienst ein eigenes Schlüsselpaar aus Public und Secret Key, indem er den geheimen Passkey mit dem Domain-Namen kryptografisch verknüpft (als Keyed-Hash Message Authentication Code, etwa HMAC-SHA256). Das bedeutet, dass der für Google verwendete Schlüssel ein anderer ist als der für

Microsoft oder Heise. Somit kann man Nutzer nicht über ihre Anmelde-Credentials tracken.

Den Public Key hinterlegt der Nutzer bei der Registrierung auf dem Server; den geheimen Passkey bekommt der Dienstbetreiber jedoch nie zu sehen – und kann ihn sich folglich auch nicht mehr stehlen lassen. Für die Anmeldung schickt der Server eine Challenge an den Authenticator. Der signiert diese mit dem passenden geheimen Schlüssel und schickt dies als Antwort an den Server. Der Anwender muss diesen Vorgang lediglich autorisieren – etwa durch das Berühren eines Sensors auf dem Token (User Presence).

In fortgeschrittenen Szenarien muss er seine Berechtigung etwa durch Eingabe einer PIN, Gesichtserkennung oder Fingerabdruck gegenüber dem Authenticator nachweisen (User Verification, UV). Dieser übermittelt dem Server lediglich das Ergebnis des UV-Tests; keinesfalls gehen dabei PINs oder gar biometrische Daten über die Leitung. Die gelegentlich anzutreffende Aussage, man melde sich mit seinem Fingerabdruck oder Gesicht an, ist also falsch oder zumindest irreführend verkürzt. Die Authentifizierung beim Server erfolgt mit dem Passkey, den man mit seinen biometrischen Merkmalen lokal freischaltet.

Das Challenge-Response-Verfahren verhindert herkömmliches Phishing mit nachgemachten Seiten und auch Replay-Angriffe sind nicht mehr möglich. Aber auch Echtzeit-Phisher mit Tools wie Evilginx2 bleiben außen vor: Die könnten zwar einen Anwender auf die Phishingseite www.he1se.de locken. Aber weil die Authentifizierung jetzt Domain-abhängig erfolgt, hilft ihnen das, was sie dort ergaunern, nicht beim Zugriff auf den echten Server www.heise.de.

Die Bausteine

Das Ganze steht und fällt mit der Verfügbarkeit des Authenticators, der die ganzen Kryptooperationen abwickeln muss. Im ersten Schritt wurde er zusammen mit dem Passkey in

einen externen Security Key in Form eines Tokens gepackt. Der Browser kommuniziert dann über das Client to Authenticator Protocol (CTAP) etwa via USB mit diesem Token. Neuere Token unterstützen teilweise auch NFC; Bluetooth hingegen hat sich dafür nicht bewährt. Der geheime Schlüssel lässt sich nicht auslesen und verlässt dieses Token nie. Muss er auch nicht, denn alle kryptografischen Operationen damit erledigt das Token selbst. Somit kann nicht einmal eine Schadsoftware auf dem PC den Passkey aus- oder mitlesen.

In der nächsten Stufe haben die Betriebssysteme diese Authenticator-Funktion direkt eingebaut. So können Windows-PCs, Macs, Android-Smartphones und iPhones mittlerweile direkt als Authenticator agieren – ganz ohne externe Token. Den Passkey speichern sie dabei wo möglich in ihrem sicheren, nicht auslesbarem Speicher (bei Apple die Secure Enclave, bei Windows möglichst im TPM). Lediglich Linux-Desktop-Systeme können da noch nichts vorweisen, hier müssen dann reine Softwarelösungen einspringen. So kann man gemäß FIDO-Spec einen Authenticator auch als Browsererweiterung realisieren. Google etwa hat das in einigen Versionen des Chrome-Browsers eingebaut.

Der eigentliche Anmeldevorgang findet zwischen Browser und dem Webserver des Dienstes statt; im FIDO-Sprech ist Letzterer die Relying Party. Die beiden kommunizieren dabei über das Protokoll WebAuthn. Praktisch alle modernen Browser beherrschen das mittlerweile; auf der Serverseite sieht das allerdings noch eher dünn aus. Doch das ändert sich gerade.

Der Authenticator errechnet für jede Domain aus dem Passkey

einen eigenen Secret Key, mit dem er die Antwort auf eine Challenge des Servers signiert. Diese Antwort authentifiziert den Anwender (Abb. 2).

Abstufbare Sicherheit

Dieses Passkey-Konzept kann man in verschiedenen Szenarien und für unterschiedliche Sicherheitsanforderungen umsetzen. Ursprünglich hatte es die FIDO Alliance als Universal 2nd Factor (U2F) spezifiziert. Die Idee dabei war, dass man die geheimen Schlüssel zusammen mit der notwendigen Software für die Challenge-Response-Authentifizierung in ein externes Token packt, wo sie auch vor Trojaner-Angriffen sicher sind. Die ersten Umsetzungen waren folglich Hardwaretoken wie die von Yubico, Feitian, SoloKeys und vielen anderen.

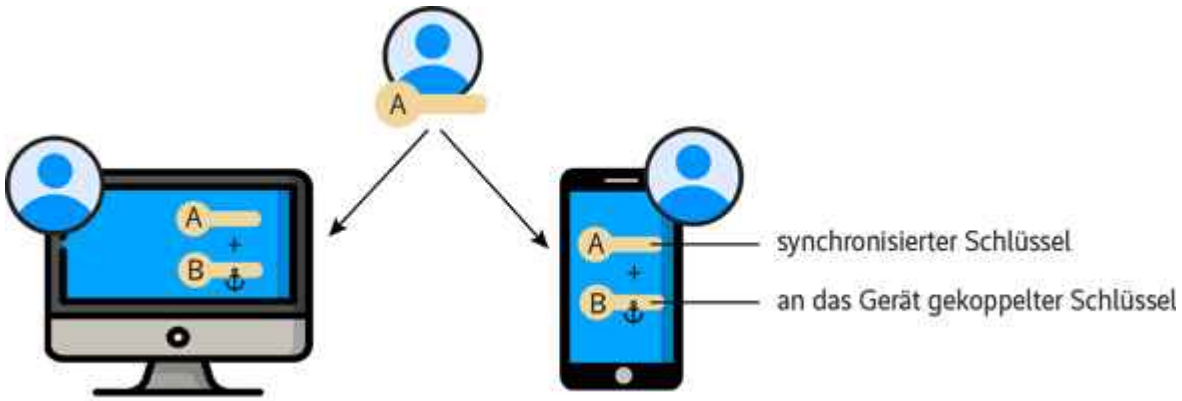
Mit FIDO2 hat die FIDO das Konzept erweitert und verallgemeinert, um Passkeys auch als einzigen Faktor zu erlauben und damit direkt das Passwort zu ersetzen, statt es nur zu ergänzen. Die Idee dabei ist: Passkeys sind auch ohne Passwörter so sicher, dass sie als alleiniger Schutz für die meisten Anwendungsfälle vollkommen ausreichen. Der Einsatz als zweiter Faktor gemäß U2F ist aber nach wie vor möglich und Teil der Spezifikation von FIDO2. Prinzipiell kann bei FIDO2 der Authenticator übrigens auch den bei einem Dienst verwendeten Benutzernamen speichern. Damit reduziert sich das sichere Anmelden im Wesentlichen auf einen Klick und den Blick in die Kamera. Praktisch wird das jedoch bisher kaum genutzt.

Trotzdem setzte sich Passkey-Authentifizierung immer noch nicht auf breiter Front durch. Das Problem: Die für höchste Sicherheitsansprüche geforderte Versiegelung der Passkeys in nicht auslesbaren Speicherbereichen verhinderte Backups und die Nutzung eines Passkeys auf mehreren Geräten. Man meldete sich mit seinem Smartphone ganz toll und komfortabel ohne Passwort bei einem Shop an, konnte dann aber auf dem PC nicht darauf zugreifen – oder umgekehrt. Und wenn man den hinterlegten Authenticator verlor, hatte man sich ausgesperrt.

Synchron via Cloud

Das soll sich jetzt ändern. Mit einer 2022 gestarteten Initiative wollen FIDO, Apple, Google und Microsoft einen weiteren Stein aus dem Weg räumen: Die Passkeys sollen übertragbar werden. Beziehungsweise noch besser: Die Geräte eines Anwenders synchronisieren sich automatisch über die Cloud. Das ist wohlgemerkt optional. Ein Server kann auch weiterhin auf Passkeys bestehen, die fest an ein Gerät beziehungsweise Token gekoppelt und damit vor dem Zugriff durch Malware geschützt sind.

Am weitesten ist dabei aktuell Apple, wo die Cloud-Vision für Passkeys in iOS und macOS bereits weitgehend umgesetzt ist. So aktivierte ich kürzlich auf meinem iPhone endlich die Zwei-Faktor-Authentifizierung für meinen Bitwarden-Account. Ich klickte lediglich: „Ja, Passkey benutzen“, bestätigte meine Identität mit FaceID und der zweite Faktor war aktiv. Mit etwas flauem Gefühl versuchte ich dann, vom MacBook aus darauf zuzugreifen. Doch siehe da: Das klappte auf Anhieb. Ich bestätigte mit meinem Fingerabdruck, dass ich berechtigt bin, den Passkey zu nutzen (UV) – und schwups war ich drin. Apple hatte den Passkey bereits über die iCloud synchronisiert.



Mittlerweile lassen sich Passkeys über die Cloud zwischen mehreren Geräten synchronisieren (Abb. 3).
Google hat das zumindest für Android 9+ und Chrome bereits ähnlich umgesetzt; dabei läuft die Synchronisierung über den Google Password Manager. Microsoft arbeitet auf Hochtouren daran, das ebenfalls noch dieses Jahr auf die Straße zu

bekommen. Da wird es von beiden Herstellern im Lauf des Jahres noch größere Ankündigungen geben. Meine ursprüngliche Befürchtung, dass sich die Konzerne dabei selbst ebenfalls Zugriff auf die Passkeys der Anwender einräumen, hat sich übrigens nicht bestätigt: Apple synchronisiert die Passkeys Ende-zu-Ende-verschlüsselt. Das bedeutet in dem Kontext, dass für den Zugriff immer mindestens ein Geheimnis erforderlich ist, das nur der Anwender, nicht aber Apple im Zugriff hat. Google folgt offenbar diesem Beispiel, und wie es aussieht, wird auch Microsoft hier E2E-Technik einsetzen.

Schöne passwortlose Welt

Damit ist die Authentifizierung mit Passkeys endlich nicht nur viel sicherer, sondern auch komfortabler als mit Passwörtern. Man meldet sich bei einem Dienst an, bestätigt, dass man den Passkey benutzen will, und kann sich künftig auf all seinen Geräten dort anmelden. Dazu muss man lediglich das jeweilige Gerät entsperren können (UV mit PIN, Fingerabdruck oder Gesicht) und das erledigt dann den Rest. Passwort braucht man keines mehr. Aber ja, Sie haben recht: Das ist zu schön, um wahr zu sein.

Ich habe mir nämlich letztlich mit meinem 2FA-Experiment bei Bitwarden doch ins Knie geschossen. Denn ich konnte zwar von iPhone und Mac aus problemlos auf das Bitwarden-Konto zugreifen. Doch auf meinen Linux- und Windows-Rechnern war ich ausgesperrt. Für die musste ich dann doch noch einen TOTP-Authenticator registrieren.

Und das wird auch noch auf absehbare Zeit hässlich bleiben. Denn die Passkey-Konzepte der Hersteller spezifizieren lediglich eine Synchronisierung im eigenen Cloud-Kontext. Als Workaround können Roaming Authenticators dienen, bei denen etwa ein iPhone eine Cross-Device-Authentifizierung für einen PC durchführt; die Kommunikation zwischen PC und Smartphone erfolgt dabei via Bluetooth.

Echte Passkey-Brücken zwischen Microsofts Azure, Googles Cloud Platform und Apples iCloud gibt es bislang aber nicht. Dabei wären APIs, die es etwa Drittanbietern erlauben, den Passkey-Sync über Plattformgrenzen hinweg durchzuführen, überaus wünschenswert. Zumindest Google und Microsoft haben diesbezügliche Absichten bereits anklingen lassen. Bis das spezifiziert, umgesetzt und ausgerollt ist, wird allerdings noch einige Zeit vergehen.

Trotzdem glaube ich, dass der Einsatz von Passkeys dieses Jahr einen großen Schritt nach vorne machen wird. Das größte Potenzial sehe ich dabei im Endanwenderbereich, wo allein die drei großen FIDO-Protagonisten Google, Microsoft und Apple so viel bewegen können, dass viele Dienstbetreiber ebenfalls auf den Zug aufspringen werden. Für den Einsatz in Unternehmen fehlen aktuell noch Konzepte für das unternehmensweite Passkey-Management. Dort hoffe ich vor allem auf verstärkten Einsatz von Passkeys als zweiter Faktor, wo FIDO2-Hardwaretoken deutlich mehr Sicherheit bieten als SMS oder TOTP. (odi@ix.de)

1. Quellen
2. Weitere Informationen zu Passkeys und FIDO2 sowie Evilginx2: ix.de/zfnt