What Are Passkeys? How Hardware-Bound Cryptographic Keys Replace Passwords for Good

Definitions / Last updated 2026-06-11 / https://www.scrambleid.com/learn/what-are-passkeys

In one sentence: A passkey is a phishing-resistant cryptographic credential that replaces a password, where the private key is held in a hardware-protected key store on the user's device and (typically) synced across their other devices in the same platform ecosystem.

TL;DR (canonical)

- **Definition:** Passkeys are FIDO2 discoverable credentials, a public-private key pair where the private key is never exposed to the sites or servers it authenticates to. Device-bound passkeys never leave the device's secure hardware; synced passkeys replicate (encrypted) within the user's platform account.
- **Phishing-resistant by construction.** The cryptographic ceremony binds to the relying-party origin; phishing sites cannot replay or relay it.
- **Synced or device-bound.** Synced passkeys (iCloud Keychain, Google Password Manager, Microsoft account) replicate across the user's devices. Device-bound passkeys live only on the device where they were created.
- **Cross-platform sign-in via QR.** A user on a Windows machine can sign in using an iPhone passkey via a QR code that triggers a Bluetooth-proximity ceremony.
- **Replacing passwords for consumer apps.** Major consumer platforms (Apple, Google, Microsoft, Amazon, GitHub, Best Buy, eBay, PayPal, and increasingly banks and government services) support passkey sign-in.
- **Enterprise viable.** Okta, Microsoft Entra ID, Ping, Auth0/Okta CIC and most modern IdPs support passkey authentication for workforce.

How a passkey works

Registration

1. The user is signed in to the relying party (or in a setup/account-creation flow).
2. The relying party calls the WebAuthn API to create a credential.
3. The user is prompted by their device (biometric, PIN, or security-key tap).

4. The device generates a fresh public-private key pair. The private key is stored in the device's hardware-protected key store. If the platform supports passkey sync, the private key is also synced (encrypted) to the user's platform account.
5. The public key and a credential ID are returned to the relying party, which stores them against the user account.

The user has not entered a password. The user has not chosen a memorable secret. The user has performed a biometric or PIN verification on their device.

Authentication

1. The user visits the relying party (logged out).
2. The relying party calls the WebAuthn API with a fresh challenge.
3. The user's device finds the relevant passkey (this is where "discoverable" matters, the user typically does not need to type a username) and prompts the user (biometric or PIN).
4. The device signs the challenge plus the origin and other context with the private key.
5. The signed assertion is returned to the relying party, which verifies it with the stored public key.

The user has not entered a password. The user has not received an OTP. The user has not approved a push.

Synced vs device-bound passkeys

The single most important architectural decision in passkey design is whether the credential syncs.

Property	Synced passkey	Device-bound passkey
Storage	Synced (encrypted) to platform account	Lives only on the device
Recoverability	Recover via platform account recovery	Recover via relying-party flow
Cross-device convenience	Available on all the user's platform devices automatically	Only the device where it was created
Hardware binding	Bound to platform account, accessible from any device in the ecosystem	Bound to specific hardware
Best for	Consumer apps where convenience and recovery matter most	High-assurance workforce contexts where hardware binding matters
NIST AAL fit	AAL2 properties typically; some implementations approach AAL3	AAL3 properties typically achievable
Examples	iCloud Keychain, Google Password Manager, Microsoft Authenticator passkeys	Hardware security keys (YubiKey) created with discoverable credential support; some enterprise authenticators

For consumer-facing applications (banking, retail, social), synced passkeys are the dominant choice because they preserve recovery and reduce abandonment. For high-assurance workforce contexts (admin access, regulated environments), device-bound passkeys with explicit attestation are often preferred. Most enterprises end up running a mix: synced passkeys for general workforce, device-bound for privileged.

Cross-platform sign-in (the QR ceremony)

A user on a Windows machine wants to sign in to a website using a passkey stored on their iPhone. The flow:

1. The website triggers WebAuthn on Windows.
2. Windows displays a QR code (this is FIDO's "hybrid transport" or "caBLE", cloud-assisted Bluetooth Low Energy).
3. The user scans the QR with their iPhone camera.
4. The iPhone and the Windows machine establish a Bluetooth proximity check (this is the security-relevant part, it ensures the iPhone is physically near).
5. The iPhone authenticates the user with biometric and signs the challenge with the passkey.
6. The signed assertion travels back to Windows and to the relying party.

This pattern lets passkeys work across ecosystems without requiring the user to set up multiple credentials. The Bluetooth proximity check matters because it prevents attackers from showing a QR code on a phishing site and having the user's phone authenticate them remotely.

Passkeys vs related concepts

Concept	What it is	Relationship to passkeys
Password	Shared secret typed by user	Passkeys replace passwords
FIDO2 credential	A public-private key pair under FIDO2	Passkeys are FIDO2 credentials (typically discoverable)
WebAuthn credential	A credential created via the WebAuthn API	Same thing as FIDO2 in browser context
Security key	A hardware authenticator (YubiKey, etc.)	A security key can hold a device-bound passkey
Platform authenticator	Built-in device authenticator	Most synced passkeys use platform authenticators
Phishing-resistant MFA	A property of the authentication ceremony	Passkeys are phishing-resistant
Single sign-on (SSO)	Federation pattern for accessing multiple apps	Passkeys can be the authentication method behind SSO

What passkeys solve

Problem	How passkeys solve it
Password phishing	Origin-bound cryptographic ceremony; phishing sites cannot replay
Password reuse	Unique key pair per relying party; nothing to reuse
Server-side breach exposing passwords	Server only has the public key; private key never on server
SIM swap (defeating SMS MFA)	SMS is not in the chain
MFA fatigue (push spamming)	No push-approve loop; user authenticates with biometric on their own device
Adversary-in-the-middle phishing (Evilginx-class)	Origin binding makes the relayed assertion invalid
Forgotten passwords / password resets	No passwords to forget; recovery is a separate, designable flow

What passkeys don't solve

Problem	Why passkeys don't address it
Identity proofing at registration	A passkey binds a credential to an account; not who the person is. Pair with identity proofing
Authorization	Passkeys authenticate; what the user can do is a separate decision
Recovery from total device loss	The recovery flow is the new attack surface; design it carefully (see Recovery and Fallback Playbook)
Authentication in non-browser contexts	Voice, IVR, in-person, and M2M need different ceremonies; see Omnichannel Authentication
Insider misuse	A passkey holder who acts maliciously is still authenticated; the audit trail matters

Enterprise considerations

Workforce passkey rollouts have considerations that consumer rollouts don't:

1. **Provisioning.** How are passkeys created during onboarding? Self-service after IT-issued temporary credential, or admin-driven?
2. **Attestation.** Does the enterprise require proof of which authenticator was used? Some authenticators provide attestation; others do not.
3. **Sync vs device-bound.** Synced passkeys are easier; device-bound passkeys provide stronger hardware binding. Mature enterprises use a mix based on role and assurance level.

4. **Recovery.** Help-desk-driven password resets are the largest post-passkey ATO surface. Phishing-resistant recovery is essential.
5. **Conditional access.** How do passkey signals flow into the enterprise's conditional-access engine (Entra ID Conditional Access, Okta Identity Engine, etc.)?
6. **Audit.** Authentication events need to be signed, queryable, and shippable to SIEM with the credential metadata that makes incident reconstruction possible.
7. **Compliance.** NIST AAL properties, FedRAMP IA-2, OMB M-22-09 for federal, NYDFS Part 500, PCI DSS v4.0.1 all interact with passkey choice.

Key Takeaway

A passkey is a FIDO2/WebAuthn discoverable credential, a public-private key pair where the private key is held in a hardware-protected key store on the user's device and (typically) synced across the user's devices via their platform account (iCloud Keychain, Google Password Manager, Microsoft account). Passkeys are phishing-resistant by construction (origin binding, hardware-bound private key) and replace passwords for both consumer and enterprise authentication. The key architectural choice is synced vs device-bound: synced passkeys prioritize convenience and recovery; device-bound passkeys provide stronger hardware binding for high-assurance contexts. Both are FIDO2-compliant and both are phishing-resistant. The biggest deployment concern is the recovery flow; help-desk-driven recovery is the largest post-passkey ATO surface.

FAQ

What is a passkey?

A passkey is a phishing-resistant cryptographic credential that replaces a password. Technically, a passkey is a FIDO2 discoverable credential where the private key is held in a hardware-protected key store on the user's device. When a user signs in, their device proves possession of the private key by signing a challenge bound to the website's origin. The website verifies the signature with the public key it stored at registration.

Are passkeys safer than passwords?

Yes, in multiple distinct ways. Passkeys cannot be phished (the cryptographic ceremony binds to the relying-party origin). They cannot be reused across sites (a unique key pair per relying party). They cannot be guessed or brute-forced (cryptographically random). They cannot be stolen from a server breach (the server only has the public key, which is harmless). And they cannot be relayed by an adversary-in-the-middle.

What's the difference between a synced passkey and a device-bound passkey?

A synced passkey is stored encrypted in the user's platform account (iCloud Keychain, Google Password Manager, Microsoft account) and replicated to all the user's devices in that ecosystem. A device-bound passkey lives only on the device where it was created. Synced passkeys are more recoverable; device-bound passkeys are more tightly bound to specific hardware. Both are FIDO2-compliant and both are phishing-resistant. The choice is about recovery model and the assurance level required.

Do passkeys work across iPhone, Android, and Windows?

Yes, with caveats. Each platform syncs passkeys within its own ecosystem (Apple, Google, Microsoft). Cross-platform sign-in works via QR-code-driven cross-device authentication: a Windows machine can use an iPhone passkey by displaying a QR code that the iPhone scans, with the authentication happening on the iPhone over Bluetooth proximity. Cross-platform passkey sync via standards (e.g., emerging credential exchange formats) is in progress but not universal as of mid-2026.

Can I use passkeys for workforce / enterprise authentication?

Yes. Enterprise IdPs (Okta, Microsoft Entra ID, Ping, Auth0/Okta CIC) support passkey authentication. The enterprise considerations are about the credential lifecycle (how passkeys are provisioned, attested, and recovered), the assurance level required (some enterprises prefer device-bound passkeys for higher-assurance functions), and integration with existing IAM (group membership, MFA policies, conditional access).

What happens if I lose all my devices?

For synced passkeys, recovery typically goes through the platform account recovery (Apple ID recovery, Google account recovery, Microsoft account recovery), which has its own multi-factor flow. For device-bound passkeys, the relying party's recovery flow is the path; ideally that flow is itself phishing-resistant. The recovery path is a critical part of the passkey design and is often where security vendors differentiate. See Recovery and Fallback Playbook.

Are passkeys phishing-resistant?

Yes. Passkeys are FIDO2/WebAuthn credentials, which are phishing-resistant by construction: the cryptographic ceremony binds to the relying-party origin, and the private key remains in the user's hardware-protected key store. A phishing site at a different origin cannot make the user's authenticator sign a valid challenge for the real origin. CISA's phishing-resistant MFA fact sheet recognizes FIDO2/WebAuthn as meeting the bar.

References (public)

- FIDO Alliance, Passkeys: https://fidoalliance.org/passkeys/
- W3C WebAuthn Level 2: https://www.w3.org/TR/webauthn-2/
- Apple, About the security of passkeys: https://support.apple.com/en-us/102195
- Google, About passkeys: https://safety.google/authentication/passkey/
- Microsoft, Passkey support: https://learn.microsoft.com/en-us/entra/identity/authentication/concept-authentication-passkeys
- CISA, Implementing Phishing-Resistant MFA: https://www.cisa.gov/sites/default/files/publications/fact-sheet-implementing-phishing-resistant-mfa-508c.pdf

Related reading

- What Is FIDO2?
- What Is Phishing-Resistant MFA?
- What Are NIST AAL Levels?
- Phishing-Resistant Web Authentication
- Recovery and Fallback Playbook