Best Practices

Yubico

May 13, 2024

CONTENTS

CHAPTER

ONE

INTRODUCTION

This collection of guides is intended to help service providers, organizations and individuals make good decisions about supporting, requiring, and using the protocols that the YubiKey supports. It includes frequently asked questions, best practices, and reference architectures.

1.1 Audiences

This guidance is broken down into three main sections, each intended for a different audience:

- Service Providers
- Organizations
- Individuals

The main difference for what these audiences require generally comes down to what part of the authentication design and authenticator selection processes they have control over. Service providers typically only control the design of the service or relying party, but they may be able to encourage use of a specific authenticator (by providing free authenticators to customers). They usually lack the authority to require a specific authenticator or even a specific authentication method. Organizations, on the other hand, typically control all aspects of authentication, from the authenticator selection to the relying party or service configuration. Finally, individuals typically only control the authenticator they use for personal use and what services they use it with.

While there are some exceptions to these situations in the form of laws or regulations around certain industries or types of services, those exceptions tend to make the decision making process easier by removing some choice.

Not all types of guidance will be applicable to all audiences, and in some cases (like the *Passkey Frequently Asked Questions*), guidance may be applicable to all audiences.

1.2 Passkeys

Before the introduction of synced passkeys, the choices for modern phishing-resistent authentication were limited to two options: use a FIDO2 hardware security key or a platform authenticator (such as Windows Hello for Business or TouchID for the Mac). Nowadays, there is a much broader spectrum of authenticator choice, and with it, a more challenging set of decisions to make.

For all audiences:

- *Passkey Frequently Asked Questions*

For Service Providers:

- *Passkey Best Practices for Service Providers*

1.3 Updates

This is a living document. The computer security landscape is constantly evolving. Changes in regulations, security needs, threat actor behavior, and the technology itself all have the potential to change how the YubiKey and the protocols it supports are best used. This document will be updated periodically and represents the current consensus within Yubico. Specific recommendations may change from time to time. Consider bookmarking this documentation and returning to it periodically to ensure you're still following the most up-to-date guidance.

CHAPTER

TWO

PASSKEY BEST PRACTICES FOR SERVICE PROVIDERS

2.1 Intended audience

This documentation is intended for developers that are implementing passkey support for service providers - web sites and services (relying parties) - using a bring-your-own authenticator model. Service providers likely won't have any control over what authenticator a user prefers, and if the user's preferred authenticator isn't available for use, they may choose not to use passkeys at all. It's even possible they'll start using a competitor's product that *does* allow them to use the authenticator of their choice - for convenience, security, or even privacy reasons - so it is paramount to make sure the passkey component is done right and empowers users to make choices.

Just as any MFA is more secure than no MFA, any phishing-resistant MFA is more secure than non-phishing resistant MFA. It follows that anything which prevents a user from using their authenticator of choice to store their passkey is likely to make a site or service less secure - as long as people can opt-out of passkeys entirely.

Developers that are building a solution for employees, contractors, or perhaps VIP clients that need to protect their accounts in a specific way for risk management or compliance reasons should be more selective about passkey authenticator use. This guidance is not intended to replace guidance for situations where the developer's organization can choose the authenticator. To find out more about how to support passkeys for employees, contractors, and VIP clients, visit Yubico's glossary page for passkeys.

The goal of this documentation is to guide developers down the path of making the best decisions for their end users, ensuring that they don't unnecessarily restrict the type of authenticator or how it's used. The WebAuthn standard, upon which passkeys are based, is very permissive by default and will allow a wide range of authenticators. In short - this document is to ensure that developers don't do extra work only to aggravate end users.

2.2 Plan for user experience challenges

It's important to acknowledge that the user experience for passkeys can be very inconsistent between browsers and platforms. Different messages and user interfaces can cause confusion and may tempt developers to restrict choices to tailor the experience for users. Unfortunately, with the rapid development of passkeys and seemingly constant change to the user interface offered by platforms and browsers, providing tailored experiences can be very demanding for developers and are likely to be undercut by future changes to platforms and browsers.

To help address this, Yubico's Passkey Workshop is continually updated to include sample strategies for providing additional guidance to end users and will continue to be updated as new developments in the passkey ecosystem unfold.

2.3 Consider how users will recover from the loss of an authenticator

Because passkeys are very secure, the normal methods for creating a "backup" don't apply. Unlike passwords or TOTP seeds, you can't simply keep a copy of the information in a safe place. Some authenticators provide passkey recoverability by syncing them between devices or accessing them via a cloud service, but these synced passkeys are in turn vulnerable to loss of access to the cloud service that houses them. As of November 2023, no passkey providers support importing and exporting passkeys as a form of backup, although some do support sharing passkeys with other users of the same passkey provider.

It's essential for services to support the registration of multiple passkeys to ensure that users can recover their account themselves if they lose an authenticator or lose access to a passkey provider. In addition to simply allowing multiple passkeys on separate authenticators, indicating which passkeys are synced or backup eligible in the user interface will help users determine if they will be able to recover a passkey if they lose or damage a particular device. Moreover, always give end users an opportunity to label a passkey when it is created, and advise them that this label is to help them determine which authenticator or passkey provider holds the passkey (such as a description of the authenticator type, model or its appearance). Lastly, be sure to give users a way to delete individual entries, as this will enable users to remove passkeys for lost devices or even to revoke synced or shared passkeys if they so choose.

Implementing device attestation may also help users understand where their passkeys are stored, as long as the authenticator supports attestation (more on this below).

While synced passkeys will ensure self-service account recovery for most users in most situations, they are not a silver bullet, and you must plan for how you will handle account recovery for users that are unable to access their authenticators where their passkeys reside.

2.4 Decide when to use discoverable credentials

Discoverable Credentials (previously known as resident credentials) allow a web browser or platform to enumerate the credentials available in an authenticator for a specific web site or domain and display them to the end user in order to make logging in easier. They are especially convenient for logon flows where the user doesn't even need to enter their username, as it can be determined from the discoverable credential itself.

While the official FIDO2 definition of a passkey is a "discoverable FIDO2 credential", in practice, discoverability hasn't been a strict requirement in most passkey implementations. For instance, both Google and Apple support saving a non-discoverable FIDO2 credential in their respective passkey ecosystems.

Discoverable credentials have two main drawbacks. The first is privacy related, and the second is related to hardware FIDO2 authenticators with limited discoverable credential storage. These drawbacks may make discoverable credentials undesirable for certain services.

Additionally, discoverable credentials, as the name implies, can be enumerated by the web browser or operating system if an attacker can unlock the associated authenticator. Conversely, non-discoverable credentials don't leave any record behind if they're stored on a FIDO2 security key, even if the authenticator can be unlocked. While this may not be a consideration for most users, for some users with increased privacy requirements, this may mean the difference between being able to use a web service safely or not.

Another important aspect to note is that hardware authenticators have limited discoverable credential storage. For example, if a user is attempting to use a YubiKey 5, they will have only a maximum of 25 credentials that can be made discoverable. While it's unlikely for the average user to hit this limit, it is still possible that users may have filled up all of the discoverable credential slots on their authenticator.

Users may have a privacy or technical need for a non-discoverable credential, therefore it is important to always provide a way for the user to initiate a login via username, and perform registrations with discoverable credentials set to *preferred*. This will allow a graceful fallback for authenticators that have already exhausted their discoverable credential storage.

Services that handle potentially sensitive information *should* optionally provide a method for specifically requesting a non-discoverable credential, to accommodate users with elevated privacy needs.

For an example of how to handle both discoverable and non-discoverable credentials, see the Yubico Passkey Workshop.

2.5 Decide when to request attestation

Attestation is the only way to achieve high confidence that a given credential is device-bound, and the only way to reliably determine the type of passkey being used. Services that want to be able to provide better information to users about the passkeys that are being used, or need to be able to use information about the passkey to make risk-based decisions about passkey use, should request attestation information during registration. Only device-bound passkeys can provide meaningful attestation, and users always have the option of declining to supply the attestation information. Detailed guidance on implementing attestation is provided in more detail in the Yubico Passkey Workshop's section on attestation.

2.6 Consider approaches for detecting passkey support in the browser or platform

Websites may want to attempt to detect whether or not a browser or platform can use passkeys before showing specific user interface elements, or altering the logon flow for users depending on the level of passkey support. While support for passkeys in general can be inferred from support for WebAuthn, there are no simple, reliable ways to determine whether browsers or platforms can support a specific type of authenticator, or support specific capabilities, like user verification or discoverable credentials.

The following code snippet can be used to detect WebAuthn support in the browser, but it does not indicate that user verification, discoverable credentials, or any specific type of authenticator is available.

```
navigator.credentials &&
navigator.credentials.create &&
navigator.credentials.get &&
window.PublicKeyCredential
```

The `isUserVerifyingPlatformAuthenticatorAvailable()` static method of the `PublicKeyCredential` interface is unfortunately only occasionally helpful in determining if a browser has access to a platform authenticator, and it behaves differently on Windows and MacOS.

User agent fingerprinting may also be used to determine what sort of browser-based support is available, but it may not reliably indicate platform support. User agents can also be set by the browser, limiting the usefulness of user agent strings for determining passkey support.

The best practice for determining whether a platform supports passkeys is to first check whether it supports WebAuthn credentials via the code snippet above, and if it does, give the user an option to register a credential.

Once you know for sure (through use) that a specific device supports passkeys, consider setting a cookie so that you don't *need* to do any detection on the next visit.

2.7 Be cautious about requiring optional WebAuthn features

Sometimes, the behavior that an authenticator will exhibit changes based on how it's configured or due to a configuration change on a browser or platform. Maybe an authenticator has run out of storage, or a user hasn't configured a PIN. It's important to understand that the arguments passed to the WebAuthn API's don't always mean a credential will be created or asserted with those settings - they mean a platform will *try* to perform an action on an authenticator with those settings.

The only way to determine how a credential has been created or used is to evaluate the data that is returned as part of the WebAuthn call, compare that with your requirements, and inform the end user of the implications of the way their authenticator has processed the WebAuthn request. For examples, see *Consider how users will recover from the loss of an authenticator* and *Decide when to use discoverable credentials*

CHAPTER

THREE

PASSKEY FREQUENTLY ASKED QUESTIONS

3.1 What is a Passkey?

Passkeys are like passwords, but better. They're better because they aren't created insecurely by humans, and because they use public key cryptography to create much more secure experiences.

But passkeys aren't a new thing. It's just a new name starting to be used for WebAuthn/FIDO2 credentials that enable fully passwordless experiences. These types of credentials are also called discoverable credentials, or sometimes resident credentials.

We like the new term and will use it, because it helps people understand they're a password replacement with a simple term. "Passkey" is much more understandable by most people than "discoverable WebAuthn/FIDO credential."

The first public mention of the term passkey to a wide audience was by Apple at a WWDC2021 talk where they introduced a "Passkeys in iCloud Keychain" technology preview to developers.

Passkeys refer only to WebAuthn/FIDO credentials. This does not include the many other keys and protocols, such as PIV, OTP, or OpenPGP Card, that are available in the YubiKey 5 Series.

3.2 Is 'passkey' the new name for FIDO and WebAuthn credentials?

Passkey is a term that the industry is rallying around for FIDO credentials that can fully replace, rather than only augment, passwords. These are called resident or discoverable credentials in the specs. We think "passkey" is a better term than "discoverable WebAuthn/fido credential," because it evokes its ability to replace passwords in an accessible way.

Passkeys in YubiKeys have been supported since discoverable credentials were added in the WebAuthn/FIDO standards around 2018. However, it's important to note that passkeys in YubiKeys are not copyable, meaning the passkey is bound to the YubiKey.

See below question: "*How are passkeys different from YubiKeys?*" for additional information.

3.3 Why is the term passkey in the news a lot recently?

Some Platform/OS vendors started shipping support for fully passwordless experiences using external authenticators like YubiKeys (and also using the security-focused hardware built into their devices, such as TPMs) as early as 2019.

Since early 2023, many Platform/OS vendors and service providers have added support passkeys. Password managers have also added support for storing a using passkeys.

Expect to see a lot more about passkeys from platform vendors such as Apple, Google, and Microsoft, as well as from external authenticator vendors such as Yubico, in the news as the implementations evolve.

3.4 How are passkeys different from YubiKeys?

Yubikeys can **contain** passkeys.

YubiKeys have had the ability to create these passwordless-enabled FIDO2 credentials (passkeys) since the YubiKey 5 Series became available in mid-2018. Currently, YubiKeys can store a maximum of 25 passkeys. We are evaluating increasing this in the future because of the likely increase in fully passwordless experiences across the web that require them.

They're different because copyable passkeys aren't stored on dedicated hardware and will be automatically synced using the credentials for the underlying cloud account, whereas passkeys in YubiKeys are bound to the YubiKey's physical hardware where they can't be copied.

3.5 What terms will Yubico use to talk about passkeys?

We like the term passkey and plan to use it. Because many things are being talked about at the same time, we will try to use terminology consistently to make the differences or similarities clear depending on the situation. This is still a work in progress across the industry, and we will adapt as things change.

The first differentiator between different types of passkeys is whether they can be copied or synchronized. These copyable passkeys are often called "multi-device," "syncable," "backup enabled," "shareable," or similar terms. We prefer to use "copyable" because it clearly describes what can be done with the credential, but it does not imply any goodness or badness and does not use overloaded or confusing terms.

We prefer to use "device-bound" to describe passkeys that can't be copied, because it aligns with the terminology the rest of the industry is using to describe passkeys. Device-bound passkeys are tied to a specific device and can't be copied or synchronized.

Once you know a passkey is device-bound, the next step is describing what kind of device it's bound to. Some device-bound passkeys are bound to general purpose computing devices like a smartphone, a laptop, or even a desktop computer. A passkey stored on a YubiKey, on the other hand, is device-bound to a portable, purpose-built security device: a security key.

Some of these terms are easily confused with the WebAuthn/FIDO concept of an authentication device's "attachment", which can have the values "platform" or "cross-platform." These terms describe how the authenticator device is attached to the system and provide a way for a web site to tell a browser where to look for a passkey, but they don't reveal anything about the passkey itself.

3.6 What are the security tradeoffs between copyable and device-bound passkeys?

Device-bound passkeys on portable, purpose-built security devices like YubiKeys are the "gold standard" for modern, phishing-resistant authentication and security. They are very easy to reason about and build systems around; no device, no access. However, for consumers registering credentials to many sites, managing multiple authenticators so you have an up-to-date backup can present challenges.

Copyable passkeys can make it easier to recover an account in the event of a lost device (as long as the user can obtain another device that works with the cloud syncing service they used). Using that copyable credential proves that there was access to a device which was logged into the user's cloud account. This can be a useful additional signal, but it does not provide the same level of security as a device-bound passkey.

3.7 How can organizations tell what type of passkeys are used to authenticate to their services?

Security keys like the YubiKey are capable of providing attestation information during registration. Services that process and store attestation information can determine information about the manufacturer, capabilities, and certifications of the security key that created a passkey. This information can help service providers detect counterfeit devices or provide guidance to users about how their passkey is stored. For more detailed information about how to handle attestation, see the Yubico Passkey Workshop's section on attestation.

3.8 What is Yubico's overall guidance about passkeys?

- We hope that a consumer focused push about passkeys will entice more services to enable support for WebAuthn/FIDO.
- Copyable passkeys offer roughly the same security as "Sign-in with Google/Apple," plus an additional key sync password.
- Today, banks, enterprises, and those wanting or needing high security do not rely solely on the security of cloud accounts provided by Sign-in with Google/Apple via federated login protocols like SAML, OpenID Connect, or OAuth. Even if copyable passkeys are used to provide that association instead, the security provided will still be insufficient for high security needs.
- The multitude of high security use cases faced by many organizations need more protocols than just FIDO. These organizations need the security guarantees and cryptographic attestations provided by hardware backed credentials to know their systems are safe and to be able to prove it.
- Attestation is also the only way to achieve high confidence that a given credential is device-bound and stored on purpose-built hardware.
- Services should continue to request, store, and use attestation information to make risk decisions based on the type of credential that is used. Our guidance on attestation is provided in more detail on our developer site.
- More use of WebAuthn/FIDO hopefully means that eventually fewer people will use and fewer services will have to deal with creating and securing dangerous username and password-based systems.

We are happy that the standards we co-created and have worked on improving for years are seeing even wider adoption, and we are hopeful that these motions will continue to reduce harm and advance our mission to make the internet safer for all.

For more specific passkey guidance for service providers, see *Passkey Best Practices for Service Providers*.

3.9 Can passkeys replace a password as well as another authentication factor?

Absolutely, yes! Passkeys have been described as a “password replacement”, which is true, but it frequently misses that passkeys can also replace the push notification, MFA code, or SMS notification that are often used to bolster password security.

Passkeys combine two authentication factors. The first is always *something you have*, which is the passkey itself.

The second may be one of:

- Something you know, which is a PIN, or a device passcode.
- Something you are, measured by a biometric sensor like a fingerprint sensor or FaceID.

The combination of these factors, as well as the phishing-resistant nature of FIDO2/WebAuthn, make passkeys more secure than passwords combined with traditional MFA.

CHAPTER

FOUR

COPYRIGHT

4.1 Trademarks

4.2 Disclaimer

4.3 Contact Information

Yubico AB
Kungsgatan 44
111 35 Stockholm
Sweden

More options for getting touch with us are available on the Contact page of Yubico's website.

4.4 Document Updated

2024-05-13 21:58:22 UTC