Best Practices

Yubico

Aug 07, 2026

CONTENTS

1 Introduction **1**
- 1.1 Audiences . . . 1
- 1.2 Passkeys & FIDO2 . . . 2
- 1.3 FIDO metadata & attestation . . . 2
- 1.4 Updates . . . 2

2 Attestation and Authenticator Metadata **3**
- 2.1 Intended audience . . . 3
- 2.2 Background . . . 3
- 2.3 AAGUIDs . . . 3
- 2.4 FIDO Metadata Service (MDS) . . . 3

3 Passkey Frequently Asked Questions **5**
- 3.1 Intended audience . . . 5
- 3.2 What is a passkey? . . . 5
- 3.3 Is ‘passkey’ the new name for FIDO and WebAuthn credentials? . . . 5
- 3.4 Why is the term passkey in the news a lot recently? . . . 6
- 3.5 How are passkeys different from YubiKeys? . . . 6
- 3.6 What terms will Yubico use to talk about passkeys? . . . 6
- 3.7 What are the security tradeoffs between copyable and device-bound passkeys? . . . 7
- 3.8 How can organizations tell what type of passkeys are used to authenticate to their services? . . . 7
- 3.9 What is Yubico’s overall guidance about passkeys? . . . 7
- 3.10 Can passkeys replace a password as well as another authentication factor? . . . 8

4 Passkey Best Practices for Service Providers **9**
- 4.1 Intended audience . . . 9
- 4.2 Plan for user experience challenges . . . 9
- 4.3 Consider how users will recover from the loss of an authenticator . . . 10
- 4.4 Decide when to use discoverable credentials . . . 10
- 4.5 Decide when to request attestation . . . 11
- 4.6 Consider approaches for detecting passkey support in the browser or platform . . . 11
- 4.7 Be cautious about requiring optional WebAuthn features . . . 12

5 Authenticator Enforcement Best Practices for Identity Providers **13**
- 5.1 Intended audience . . . 13
- 5.2 Overview . . . 13
- 5.3 MDS Management . . . 14
- 5.4 Customer-centric enforcement . . . 14
  - 5.4.1 Do not override customer authenticator selection . . . 14
  - 5.4.2 Allow custom metadata . . . 14

5.5 Use cases . . . 15
5.5.1 Non-certified devices with MDS metadata . . . 15
5.5.1.1 Authenticator verification and testing . . . 15
5.5.1.2 Vulnerability remediation . . . 15
5.5.2 Customer-provided metadata . . . 15
5.5.2.1 Custom authenticators . . . 15
5.5.2.2 Certified devices not listed in MDS . . . 16
5.6 Considerations for authenticator selection . . . 16
5.6.1 Security vetting and assessment . . . 16
5.6.2 Additional considerations . . . 16
5.7 IdP implementation recommendations . . . 17
5.7.1 Lifecycle and MDS updates . . . 17
5.7.2 Logging and API access . . . 17
5.7.3 Known incompatible devices . . . 17

6 Glossary **19**

7 Copyright **21**
7.1 Trademarks . . . 21
7.2 Disclaimer . . . 21
7.3 Contact Information . . . 21
7.4 Document Updated . . . 22

Index **23**

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 collection of guides is made up of a variety of different kinds of articles, from conceptual articles that lay a foundation for understanding how a technology works to FAQs and prescriptive best practices guidance.

While we expect readers to be familiar with the *kind* of article they're reading, there's a lot of content in this Best Practices book that may seem inconsistent or even conflicting if the intended audience isn't made very clear. We've made sure to label each article with its intended audiences and, when applicable, a short discussion of *why* the guidance applies to each of them.

We have identified four main audiences:

- Service Providers (individual or customer identity)
- Identity Providers (organization or workforce identity)
- 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. Individuals, on the other hand, typically only control the authenticator they use for personal use and what services they use it with.

Identity providers and organizations tend to have closer coordination, and together control all aspects of authentication, from the authenticator selection to the relying party or service configuration, and can enforce those decisions with either technology or policy controls. Organizations may even build their own relying parties for authentication - in which case they'll act as the identity provider as well.

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 & FIDO2

Before the introduction of synced passkeys, the choices for modern phishing-resistant authentication were limited to two options: use a FIDO2 hardware security key or a platform authenticator (such as Windows Hello for Business or Touch ID 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 FIDO metadata & attestation

The FIDO specifications allow for various levels of privacy regarding individual FIDO authenticators. This guidance discusses the tradeoffs made between privacy and compliance for passkeys and contains specific recommendations for Identity providers processing attestation metadata and for organizations navigating the attestation ecosystem.

Concepts:

- *Attestation and Authenticator Metadata*

For identity providers:

- *Authenticator Enforcement Best Practices for Identity Providers*

1.4 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

ATTESTATION AND AUTHENTICATOR METADATA

2.1 Intended audience

This conceptual article will be of most interest to organizations that need to ensure that their workforce is using a strong authenticator and identity providers looking to provide that assurance to organizations via their products.

2.2 Background

Organizations may want to limit the use of FIDO authenticators to specific makes and models to ensure the devices users authenticate with meet their security and compliance needs. In order to facilitate this, the FIDO specification provides a mechanism called *attestation*. Attestation uses data supplied by device manufacturers and validated and aggregated by the FIDO Alliance to verify cryptographic signatures generated during FIDO authenticator registration.

2.3 AAGUIDs

Identity providers (IdPs) use an authenticator's *AAGUID* to look up a certificate chain that can be used to validate the attestation statement's signature. IdPs may also use the AAGUID to look up other information about an authenticator, such as the features it supports, or the presence of specific 3rd party certifications, or even an icon to display to help a user to determine where their passkey is stored.[1] This information is collectively known as the *authenticator metadata*.

2.4 FIDO Metadata Service (MDS)

In order to facilitate IdPs' use of *authenticator metadata*, the FIDO Alliance hosts the FIDO Metadata Service (*MDS*), which is updated as vendors produce new authenticators with different *AAGUIDs* and as authenticators achieve FIDO and 3rd party certifications. The *MDS* prevents IdPs from having to hard code AAGUID and certificate chain information.

The *MDS* is a single large document which is over 5 megabytes in size at the time of writing. The bandwidth and compute requirements to obtain and process it make it unsuitable for checking during every new authenticator registration. IdPs should download and ingest a new MDS document into their systems periodically, out of band from the authenticator registration process.

The FIDO Alliance recommends that the MDS is downloaded and ingested monthly.[2] However, in 2024, the MDS averaged between one and two updates *per week*. Identity providers may want to download and process the MDS more frequently to support the rapid changes in the FIDO authenticator ecosystem.

[1] FIDO Metadata Statement standard

[2] "How often should I be fetching MDS3 blob?" at https://fidoalliance.org/metadata/

The MDS is not exhaustive. There is no requirement to list an authenticator in the MDS, even if it has received a FIDO Alliance certification. There are a number of reasons why device manufacturers or customers may not want the metadata for their authenticators published, but ultimately customers must have the option of privacy, even if they are using an authenticator that has been customized for them by a device manufacturer.[3]

Synced passkey providers aren't even eligible for listing in the MDS because, with some exceptions, they are not currently capable of producing attestations.

More information about the MDS can be found at the FIDO Alliance Metadata Service page: https://fidoalliance.org/metadata/

[3] See the *Use Cases* section in the "Authenticator Enforcement Best Practices for Identity Providers" chapter of this guide for more detail.

CHAPTER

THREE

PASSKEY FREQUENTLY ASKED QUESTIONS

3.1 Intended audience

This FAQ is for everyone because passkeys are for everyone. Whether you're looking to understand what passkeys mean to you as an individual or for your organization, you consume identity information (relying parties) or produce it (identity providers), or you just want to log on to your favorite web site, this FAQ will help you understand what passkeys are and some of the terms used to describe them.

3.2 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.

3.3 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.

3.4 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.

3.5 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 100 passkeys.

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 under normal circumstances.

3.6 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 under normal circumstances, 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 under normal circumstances.

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.

3.7 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.

3.8 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.

3.9 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*.

3.10 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

PASSKEY BEST PRACTICES FOR SERVICE PROVIDERS

4.1 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.

4.2 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.

4.3 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.

4.4 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 for YubiKeys with firmware 5.0 - 5.6.x or 100 credentials for YubiKeys with firmware 5.7+. 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.

4.5 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.

4.6 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.

4.7 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

FIVE

AUTHENTICATOR ENFORCEMENT BEST PRACTICES FOR IDENTITY PROVIDERS

5.1 Intended audience

This documentation is intended for identity providers and organizations who would like to limit authenticator selection for self-service FIDO authenticator enrollment within a bring-your-own authenticator model.

Service providers likely won't have any control over which authenticator a user prefers, but identity providers that facilitate access to a specific organization will likely need to accommodate organization preferences or hard requirements for specific authenticator models.

This document will outline the steps that identity providers can take to allow customers to limit end user authenticator choice while still providing the flexibility required to support custom authenticators, preview devices, technology pilots, and vulnerability remediation.

5.2 Overview

Organizations may wish to limit FIDO authenticator use to specific makes and models to meet their security and compliance requirements. The FIDO specification supports this through a mechanism called attestation, which uses manufacturer-supplied data, aggregated in the FIDO Alliance's Metadata Service (MDS)[1] , to verify cryptographic signatures during authenticator registration.

Identity Providers use an authenticator's *AAGUID* to identify which public keys from the MDS can validate attestation signatures. IdPs may also use the MDS to retrieve additional information, such as supported features, FIDO Alliance[2] or third-party certifications.

For a conceptual overview of how attestation and the FIDO Metadata Service work together, see *Attestation and Authenticator Metadata*

[1] FIDO Metadata Overview https://fidoalliance.org/metadata/
[2] FIDO Certification Levels https://fidoalliance.org/certification/authenticator-certification-levels/

5.3 MDS Management

The *MDS* is a large document[Page 13, 1], making it unsuitable for real-time validation during every new user registration. Instead, the FIDO Alliance recommends that IdPs download and ingest updated MDS blobs at least once per month[3]. This ensures access to the latest device information, certifications, and security features. However, the MDS is not comprehensive. Even certified authenticators are not required to be listed. This omission may reflect privacy preferences of manufacturers or customers, particularly when devices are customized.

5.4 Customer-centric enforcement

5.4.1 Do not override customer authenticator selection

While it is appropriate to offer default minimum requirements for authenticator selection, customers must always have the final say in which authenticators are approved for use in their environment. IdPs should avoid rigidly enforcing any default minimum requirements they impose.

Rigidly enforcing default minimum authenticator requirements may:

- Prevent customers from testing emerging features or newly developed devices.
- Delay deployment of devices needed to address vulnerabilities.

Instead, IdPs should warn customers when their selected authenticators do not meet default criteria, but still allow their use. This provides flexibility for organizations to align device selection with their threat models and risk tolerances.

5.4.2 Allow custom metadata

Because the MDS does not include every authenticator model, IdPs must support customers' ability to add manufacturer-provided metadata to allow enforcement for such devices. At a minimum, this includes:

- The device's AAGUID
- A list of valid trust anchors for attestation

Note: An authenticator's absence from the MDS does not imply insecurity. Privacy is often the primary reason for non-listing.

Examples of custom metadata needs include:

- Custom AAGUIDs: Large organizations may request custom identifiers from manufacturers without disclosing this publicly.
- Co-development projects: Devices under joint development may be used internally before public release or certification.

Because custom metadata is sensitive, IdPs must maintain robust audit logs for its addition or removal and notify customers of any conflicts between custom and official MDS entries.

[3] "How often should I be fetching MDS3 blob?" at https://fidoalliance.org/metadata/

5.5 Use cases

New AAGUIDs are issued when an authenticator's functionality or metadata changes[5] . These identifiers enable fine-grained control during enforcement, even for devices that are uncertified. Organizations may encounter legitimate use cases more frequently now that FIDO adoption has accelerated than they ever have before. This section groups related use cases into two broad classes: devices which are not certified but are listed in the MDS and devices which are not listed in the MDS but may or may not be certified.

5.5.1 Non-certified devices with MDS metadata

These are authenticators for which manufacturers have published metadata in the MDS but have not yet completed certification. Examples include:

- Preview units sent for evaluation
- Production devices released before certification

Preview devices may never be certified due to cost or timing constraints. While production models are usually certified eventually, this is not guaranteed. Organizations may also supply their own metadata to support these devices, but they should weigh the added risk from using non-FIDO-managed data.

5.5.1.1 Authenticator verification and testing

Customers may wish to evaluate uncertified authenticators for:

- Compatibility testing
- Assessment of emerging technologies
- Feature development with vendors

Allowing AAGUIDs to be added for testing purposes reduces operational friction. It enables testing without requiring the deactivation of attestation enforcement, thus maintaining security boundaries.

5.5.1.2 Vulnerability remediation

Vendors may assign new AAGUIDs to firmware versions that fix vulnerabilities. These updates may precede certification. If the threat from the vulnerability is greater than the risk of uncertified usage, temporarily allowing non-certified AAGUIDs can mitigate risk. Once certified, no configuration changes are needed.

5.5.2 Customer-provided metadata

5.5.2.1 Custom authenticators

Some organizations use devices that are not listed in the MDS. This may be due to privacy concerns or specialized requirements. Customers should have the ability to:

- Add specific AAGUIDs
- Provide custom metadata

While these devices carry added risk, customers are responsible for evaluating and accepting that risk.

[5] Yubico Support "Best practices for managing AAGUID changes" https://support.yubico.com/hc/en-us/articles/17779217014812-Best-practices-for-managing-AAGUID-changes

5.5.2.2 Certified devices not listed in MDS

Some certified devices are intentionally excluded from the MDS. IdPs should support enforcement for such devices by allowing customers to manage metadata manually.

5.6 Considerations for authenticator selection

5.6.1 Security vetting and assessment

Non-certified authenticators

Organizations should evaluate uncertified devices using internal testing, documentation reviews, and compensating controls. Review the FIDO Alliance's Level 1 certification requirements to establish baseline expectations.

Custom metadata

Custom metadata introduces potential security gaps. Most metadata elements (such as supported algorithms) can be validated using the FIDO GetInfo command via tools like libfido2[4] . However, attestation root certificates must be independently validated. Errors can result in failed authentications or vulnerability to spoofed devices. Allowing customers to validate that the selected attestation root certificate(s) are valid for the AAGUID in the custom metadata can be achieved by evaluating an attestation statement.

Non-attestable AAGUIDs

AAGUIDs that lack associated root certificates cannot be attested. In such cases, malicious actors could potentially spoof device identities. Mitigate this risk with compensating controls.

Implementing compensating controls

When using uncertified or non-attestable authenticators, apply additional safeguards such as:

- Behavioral analytics
- Risk-based authentication
- IdP-specific usage policies

5.6.2 Additional considerations

- Establish a robust firmware vulnerability disclosure process
- Regularly review custom metadata for continued relevance
- Deprecate or remove custom AAGUIDs once they appear in the MDS with appropriate certifications
- Validate AAGUID entries periodically
- Prepare migration plans for deprecating unauthorized devices
- Monitor for shifting security standards or new threats

[4] Yubico libfido2 on Github https://github.com/Yubico/libfido2/

5.7 IdP implementation recommendations

5.7.1 Lifecycle and MDS updates

To ensure continued enforcement and alignment with FIDO guidance, IdPs should:

- Download and cache the FIDO MDS blob at least monthly, as per FIDO Alliance recommendations[Page 13, 1]
  - Consider downloading and processing the FIDO MDS blob more frequently, to support the rapidly evolving FIDO authenticator ecosystem.
  - Display the current MDS version in use and the time/date of the last update
- Define clear rules for metadata precedence (MDS vs. custom)
  - Alert customers to conflicts between metadata provided by the MDS and by custom entries
  - Indicate the metadata source (MDS or custom) in user interface when displaying metadata.
- Support root certificate uploads for custom AAGUIDs
  - Warn administrators of the risks associated with using AAGUIDs missing attestation root certificate information
  - Detect and warn administrators of custom AAGUIDs with malformed or un-parsable attestation root certificate information

5.7.2 Logging and API access

Managing AAGUIDs and certificates is prone to human error. IdPs should support secure workflows by implementing:

1. Read-only API
   - Provide authenticated access to AAGUID and metadata listings
   - Support certificate validation through the API
2. Writable API
   - Enable automated metadata updates via secure, authenticated endpoints
   - Accommodate regulatory and procurement workflows
3. Change logging
   - Record all metadata changes in detail
   - Use SIEM-compatible formats for logs

5.7.3 Known incompatible devices

Some IdPs, such as Entra, require device support for specific features like hmac-secret. Metadata overrides might allow unsupported devices unless properly restricted. IdPs should:

- Display UI indicators for known incompatible devices
- Prevent customers from uploading unsupported device metadata

CHAPTER

SIX

GLOSSARY

AAGUID
: An Authenticator Attestation Global Unique Identifier (AAGUID) is a 128-bit identifier representing make and model for a group of FIDO2 authenticators that share the same *authenticator metadata*. The AAGUID allows a service provider to determine which attestation root certificate chain to treat as trusted for a specific attestation statement, without revealing information that can be used to track an individual authenticator. AAGUIDs are typically written out as a 32-character hexadecimal string, sometimes with dashes after the 8th, 12th, 16th and 20th digits. For example, the AAGUID for the YubiKey 5 NFC is d7781e5d-e353-46aa-afe2-3ca49f13332a.

Authenticator Metadata
: Authenticator metadata is information about authenticator which helps relying parties or identity providers support authenticators with different capabilities, manufactured by different vendors. This information includes the *AAGUID*, a certificate chain that can be used to validate attestation statements, and information about how the authenticator can be connected (USB, NFC or Bluetooth) as well as what extensions are supported.

MDS
: FIDO Metadata Service (MDS) is a service provided by the FIDO Alliance that defines a uniform, vendor-agnostic method for looking up FIDO *authenticator metadata* by its *AAGUID*. Additional information about the FIDO Metadata Service can be found at the FIDO Alliance: https://fidoalliance.org/metadata/

CHAPTER

SEVEN

COPYRIGHT

© 2024-2026 Yubico AB. All rights reserved.

7.1 Trademarks

Yubico and YubiKey are registered trademarks of Yubico AB. All other trademarks are the property of their respective owners.

7.2 Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design, and manufacturing. Yubico shall have no liability for any error or damages of any kind resulting from the use of this document.

Yubico Software referenced in this document is licensed to you under the terms and conditions accompanying the software or as otherwise agreed between you or the company that you are representing.

7.3 Contact Information

Yubico AB
Gävlegatan 22
113 30 Stockholm
Sweden

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

7.4 Document Updated

2026-08-07 00:43:36 UTC

INDEX

A

AAGUID, 19
Authenticator Metadata, 19

M

MDS, 19