yubico

WEBAUTHN WHITE PAPER SERIES SEPTEMBER 2019

SIM Swap: Protecting against Account Take-overs with WebAuthn

Executive Summary

This paper is the fourth in a series of WebAuthn whitepapers published by Yubico. For an introduction to WebAuthn and why it is both more secure and easier to use, see the first paper, *Introducing WebAuthn: Enabling a Streamlined and More Secure User Authentication Experience.* For a deeper dive into the problems with passwords, how WebAuthn addresses them, and how to implement WebAuthn support from a developer perspective, see the second paper, *The WebAuthn Standard: Why it Matters and How it Works.* To learn about using a security key as a portable root of trust, see the third paper, *Establishing a Secure, Portable Root of Trust with WebAuthn.*

Billions of dollars are being stolen annually due to account takeover fraud. Account takeover attacks based on taking control of a person's mobile phone number or intercepting SMS text messages are increasing. To protect users against these types of attacks, relying parties (RPs) should transition away from using mobile phone numbers for multi-factor authentication (MFA) or to send recovery access codes. While these are common practices today, it can no longer be assumed that the phone number on file is currently under the control of the account owner, and text messages with access codes are subject to phishing and man-in-the-middle attacks. RPs need to move to WebAuthn to improve authentication security. WebAuthn protects against these common mobile phone-based attacks because it is built on public key cryptography, with private keys that never leave the user's device. Transitioning completely off phone numbers might take time, but RPs can quickly update processes to mitigate this risk by giving users the option to authenticate with WebAuthn security keys and encouraging them to opt out of having their phone numbers be part of the authentication and/or recovery flows.

Introduction

If you are an architect or technical decision maker who is responsible for protecting user accounts and providing secure authentication, this guide will give you best practices on key implementation decisions to make when upgrading your app or site's protection from phone and/or text-based methods to WebAuthn.

The problem with current phone-based MFA methods

Billions of dollars are being stolen annually due to account takeover. Payment systems, traditional accounts and cryptocurrencies are under increasing attack, stemming from the fact that access to these systems is based on access to a phone number. In fact, account takeover attacks based on taking control of a person's phone number or intercepting SMS access codes increased 45% from 2016 to 2018[1].

Mobile carriers are required to allow a customer to move their phone number to other carriers; and for customers who have had phones lost or stolen, or who want to upgrade to a new phone, they offer porting of phone numbers to another device. This provides convenience for the customer, but has introduced an attack vector. A common variant of "SIM swapping" or "SIM jacking" attack occurs when an attacker convinces a victim's mobile phone carrier to port the victim's mobile phone number to a phone the attacker owns. At this point, the attacker can receive phone calls and text messages intended for the victim. The attacker will then use this to gain further access to any account that is protected using the victim's mobile phone number, from an email account to other online accounts ranging from social media to banking and cryptocurrency.

We've known for years that passwords alone are insufficient to protect accounts and that MFA is the recommended mitigation. But not all MFA solutions are equal. The recent and ongoing flood of mobile phone-based hacks demonstrates that the most common phone-based ways of securing accounts with MFA–phone calls and SMS text messages delivering one time codes–are more beneficial to attention-seeking hackers and criminal organizations. Further, sites that offer account recovery using phone or text messages enable hackers to gain access to accounts without even stealing passwords. Phone apps that receive notifications or generate one-time codes are safer, but are still susceptible to social engineering and man-in-the-middle attacks.

This doesn't mean that MFA itself is a problem, rather it is the use of mobile phone-based methods of authentication that is the problem. RPs need a strong authentication method and should enable account recovery methods that are more secure than an SMS text or or an app that provides a one-time code, one that protects users' accounts while providing a good user experience. Fortunately, a method for this exists today.

WebAuthn: The open standard for secure strong authentication

The W3C Web Authentication (WebAuthn) standard provides the safer, easier, phishing-resistant login method required to protect users from these common threats. In short, here's why:

- It's based on asymmetric cryptographic standards in which the private key never leaves the user's device. Phishers can't harvest it (it's not SIM card-based, and mobile carriers can't transfer it).
- It won't work on typo-squatting domains (that use, for example, common misspellings of the site name), so phishers can't use it for man in the middle (MITM) attacks. The authenticator will respond only to the domain that was used to register a credential.
- WebAuthn is easy to use: users plug in a hardware security key and touch it, or otherwise use one that's built into their PC or phone. A PIN or

[1] The Forter Fraud Attack Index (Sixth Edition) https://l.forter.com/forter-fraud-attack-index

a biometric sensor is used for additional security when the security key is used for passwordless authentication.

- WebAuthn works in conjunction with the FIDO Client To Authenticator Protocol version 2 (CTAP2) to securely create and retrieve credentials on a security key. The two standards work in tandem making it easier for developers as they only need to concern themselves with interfacing with the WebAuthn specification.

Integrating WebAuthn into your site or mobile app

The Yubico whitepaper *The WebAuthn Standard: Why it Matters and How it Works* details the most fundamental WebAuthn scenarios–registration and authentication–from a developer perspective, as well as the user experience. This paper will build on that foundation by providing a quick recap of those basic scenarios, then go into more detail on implementing each of these experiences and provide best practices for what to do, what to avoid, and how to give your users the best experience while protecting them.

The industry has already made great progress building WebAuthn support into the key platforms and browsers. Today, Windows and Android platforms as well as Chrome, Firefox, Microsoft Edge and Safari browsers have some form of support. Web services and identity providers such as IBM Security Access Manager (ISAM), Daon, Avatier, Nok Nok S3 Authentication Suite, SingularKey, Okta, OneLogin, Ping Identity, Google G Suite and Microsoft Office 365 using Microsoft Azure Active Directory have integrated WebAuthn support as well, and rapid progress is being made. In the interim, for platforms like iOS, Yubico offers SDK support to enable mobile apps to use security keys. So what remains for websites and apps (WebAuthn RPs) is to build experiences that enable your users to benefit from this support. Specifically those experiences are:

Register a Credential

Registration is how you enable your users to create a WebAuthn credential that they can use with your site or app. Many sites refer to this experience as “Add a security key” or something similar. Your site or app invokes this flow by sending a challenge to the WebAuthn API. Then it must validate the response and, if successful, store the public key and other response information associated with the user.

Use a Credential

Once a key is registered, the user can use it to authenticate. When this happens, your app or site must again send a challenge and other information via the WebAuthn API to the security key and validate the response, verifying that the key used to generate the response is the proper key for that user.

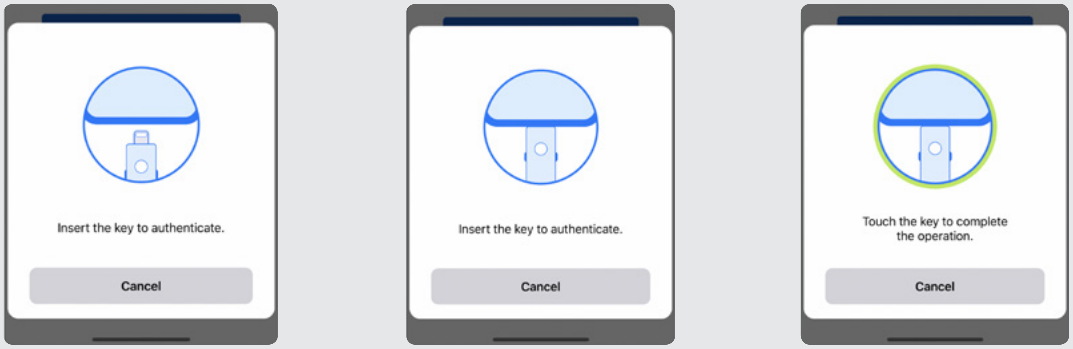
Account Recovery

Your users should be able to use security keys to recover access to the account and/or reset passwords using the WebAuthn credential. This is especially true if your app or site currently uses SMS text for these experiences.

Manage Credentials

In addition to the basic registration and authentication experiences, you’ll want to provide the following management experiences to enable your users to utilize their keys successfully:

- Manage (view, name, delete) WebAuthn credentials
- Select WebAuthn credentials as authentication, MFA, and/or account recovery factors



Insert the key to authenticate.

Insert the key to authenticate.

Touch the key to complete the operation.

Exhibit 1. Animation showing security key with a lightning connector used on an iOS device.

WebAuthn Integration Best Practices

Implementing the basic registration and authentication use cases for WebAuthn with security keys can be straightforward, but in order to provide a solution that will be adopted and used, it is important to think through all of the various user scenarios. An environment that doesn't use SMS for authentication or passwords introduces a number of different user stories that need to be understood. Having a clear plan in advance will help you ensure that there is no need to fall back on SMS-based verification solutions. Below is a set of best practices to follow when implementing security keys.

Register a Credential

Registering a security key is a simple process, but users who are new to the process will need clear instructions. As shown in Exhibit 1 above, many successful sites have created animations that show a user how to insert a security key into the port on their device, and prompt them to touch the security key when required. These types of instructions can be very helpful in guiding new users through the process.

The user should be able to register multiple security keys per account and to name each security key. This will enable the user to have a back-up key should their primary key be lost or stolen, as well as help the user identify which keys they use on the site. The name of the security key is stored by the RP and is unique to the RP. Registration timestamps and location information also help the user identify security keys that are being used.

Security keys include metadata with an attestation certificate. This information includes the manufacturer and device name, supported transports, image URL, etc. It is important to capture this information as it provides a way to validate devices, customize user experiences and provide authenticator metrics reports. This information also allows the RP to take action on an authenticator if a security vulnerability is discovered. A trust store containing all registered attestation certificates provides an audit trail to support high assurance MFA. For more information about attestation certificate please visit the *attestation section of Yubico's WebAuthn Developer Guide.*

Manage Credentials

After a security key has been registered, it is important that it can be managed. The user needs to be able to add additional security keys as well as identify those that are being used. Identification information needs to include the name of the authenticator, as well as the time and place of last use. The user should be able to rename and remove their security keys at their discretion. Users need to be given ample warning if they are removing all security keys from an account as this could leave the user without any second factor protection.

Additionally, users should be advised to opt out of having their phone number be used for authentication or verification. Removing SMS completely from your environment can take time, but users should be allowed to remove the weaker authentication option if they have setup stronger authentication mechanisms.

Step-up or Conditional Authentication

For events that carry high levels of risk or have monetary consequences, a security key can be used to quickly validate that the user is in possession of the physical security key and intends to perform the action. Prompting the user to touch the security key before proceeding allows the system to know that the request has come from the user and not from a compromised system. The user is required to provide authentication specifically for these events.

User Adoption

There are many benefits for the user in switching to a security key. They may have heard the concerns about SIM swapping but don't want to have to remember another password or PIN. They may want to switch, but might not know how to start, or are too busy to change. A little nudge at the right time will help users improve their security and online experience. Each RP will need to decide what the appropriate message is to educate the user; for example, if a user authenticates using SMS-based MFA, the RP can notify the user that stronger and more convenient authentication options are available. Additionally, the RP can query the user's browser to see if it supports WebAuthn and provide appropriate guidance and messaging.

In addition to educating users to move away from SMS-based MFA, RPs should encourage users to self-register multiple authenticators. Having multiple authenticators is the best way to deal with a lost or forgotten security key.

WebAuthn is being built into mobile devices, personal computers and, of course, security keys. This provides a wide range of options for the user to add additional authenticators. As a best practice, a security key should always be used as one of the user's multiple authenticators.

Account Recovery

Where passwords are still used, SMS is a common way to send a password reset code to the user. For a more secure account recovery, a security key can be used instead to verify the user has permission to reset the password. The *forgot password* flow would prompt the user to touch their security key instead of having a code sent. Once the user touches the YubiKey, the signed assertion public/private key pair will be validated. If valid, the user would be allowed to change the password. This process is fast and painless for the user.

If an individual loses all of their authenticators (this will become less likely the more security keys are enrolled), the identity proofing process should be performed again. Because the risk of a person losing all of their authenticators should decrease over time, it is a good practice to adopt the same rigor as for the initial access. Remote proofing technologies and processes are currently evolving in the industry and show promise to accelerate the process. In the meantime, in-person identity proofing may be required for specific and sensitive scenarios.

Portable Root of Trust

As described in the Yubico whitepaper, *Establishing a Secure Portable Root of Trust with WebAuthn,* a security key is portable and can be used across many devices for authentication, backup and bootstrapping new authenticators, and a platform based authenticator has limitations. For example, a phone based platform authenticator is convenient, but the built-in credentials on it cannot be transferred to another device when the mobile device is replaced. Using a security key as a portable root of trust makes it more convenient and affordable for users to bootstrap a new mobile device and ensure there is no disruption in access.

Mobile Scenarios

WebAuthn can be used to remove SMS-based authentication from mobile applications as well. WebAuthn support is built into Android based phones and can be leveraged by developers building mobile applications on that platform. At this time (09-2019), applications on iPhones require the integration of Yubico's iOS mobile SDK to take advantage of WebAuthn. With the YubiKey 5Ci and the iOS Mobile SDK, developers can provide apps on the iPhone with the same security and ease of use capabilities as other devices. For further details please review *Yubico's Mobile SDK.* If you have any questions concerning integrating YubiKeys into your mobile application don't hesitate to reach out to the *Yubico Developer Program.*

Conclusion and Next Steps

Removing SMS-based authentication and leveraging WebAuthn credentials will considerably improve the user's online experience while reducing the opportunity for fraud. Given the advancements in and adoption of industry standards, now is the time to implement WebAuthn solutions. Working out the user flows and adoption strategies ahead of time is critical to moving users away from SMS-based MFA solutions successfully. Yubico's Developer Program and Professional Services teams have a wealth of experience providing guidance and deploying WebAuthn solutions quickly.

Get started on your WebAuthn journey right away with the following resources from Yubico:

- Read the other white papers in the Yubico WebAuthn series:
  - *Introducing WebAuthn: Enabling a Streamlined and More Secure User Authentication Experience*
  - *The WebAuthn Standard - Why it Matters and How it Works*
  - *Establishing a Secure Portable Root of Trust with WebAuthn*
- To learn more about the WebAuthn API flows like Registration and Authentication, read our *WebAuthn Developer Guide*
- Download the *Yubikey Mobile SDK*
- Visit the *Yubico developer site* to gain access to Yubico libraries and tools, including server side libraries.

If you have any questions on this topic or have specific scenarios you want to discuss, please contact your Yubico representative for more information. You can also connect with Yubico online at *https://www.yubico.com/support/contact/.*

yubico

About Yubico

Yubico sets new global standards for simple and secure access to computers, mobile devices, servers, and internet accounts.

The company’s core invention, the YubiKey, delivers strong hardware protection, with a simple touch, across any number of IT systems and online services. The YubiHSM, Yubico’s ultra-portable hardware security module, protects sensitive data stored in servers.

Yubico is a leading contributor to the FIDO2, WebAuthn, and FIDO Universal 2nd Factor open authentication standards, and the company’s technology is deployed and loved by 9 of the top 10 internet brands and by millions of users in 160 countries.

Founded in 2007, Yubico is privately held, with offices in Sweden, UK, Germany, USA, Australia, and Singapore. For more information: www.yubico.com.