# Secure Authentication for Everyone! Enabling 2nd-Factor Authentication Under Real-World Constraints

Julian Fietkau[1(✉)], Syeda Mehak Zahra[1], and Markus Hartung[2]

[1] Technical University of Berlin, Berlin, Germany
fietkau@tu-berlin.de
[2] Avira Operations GmbH, Tettnang, Germany
markus.hartung@avira.com

**Abstract.** Millions of user accounts have been exposed by data breaches within the last years. The leaked credentials pose a huge threat to many because they can be used for credential stuffing and brute-force attacks across all online services. The best solution for this problem seems to be the use of 2nd-factor authentication, like hardware tokens or one-time passwords. While these are great solutions, they cause many problems for users because they are too expensive, difficult to manage, or just not user-friendly. In this paper, we will present the results of a study that shows that users need and want secure authentication, as long as it is quick, easy, and free of charge. Hence, we investigate how recent advancements in smartphone security and authentications standards can be used to build a mobile authenticator that is easy to use, free of charge, and as secure as a hardware token. Therefore we leverage the Trusted Execution Environment of the Android platform to implement a FIDO compliant authentication mechanism on the smartphone. Furthermore, we integrate this mobile authenticator into a password manager app, to reduce user interaction, simplify the setup and provide an encompassing solution for the user.

**Keywords:** 2nd-factor authentication · Data breaches · Leaked credentials · Fast IDentity Online · Trusted execution environments · Secure logins · Low cost security · Password manager · Biometric authentication

## 1 Introduction

The number of leaked credentials caused by data breach attacks has been increased tremendously over the past years [16,20,23]. According to our statistics, there is a steady increase in the number of breaches since 2005 and an increase of 45% just in the year 2017 [15]. Moreover, many users use the same or a similar password for every service, hence more than just the targeted service is under threat [16]. To tackle this issue, companies and researchers are trying their best to secure user logins against password stuffing attacks.


E. Gelenbe et al. (Eds.): EuroCybersec 2021, CCIS 1596, pp. 89–101, 2022.
https://doi.org/10.1007/978-3-031-09357-9_8

One of the most promising solutions to this problem is 2nd-factor authentication. Here, Users have to prove their identity twice during the login process by providing their password and a 2nd-factor such as hardware tokens, mobile TANs, or providing a cryptographic signature. The FIDO Alliance and its biggest partners, e.g. Google, are playing a key role in this fight to make authentication more secure, e.g., by promoting the use of hardware tokens [11]. Hardware tokens are small little devices, just like USB thumb drives, that can be connected to most devices to enable 2nd-factor authentication. While the cryptographic fundament is very solid and the overall idea of hardware tokens is outstanding, we suppose that hardware tokens don't scale. The reason for this assumption becomes clear, even before a user is ready to use it. The average price of a security token is somewhere between $30–$70 [25]. This price tag appears small for some people but becomes problematic if we imagine worldwide adoption. It becomes even worse when we incorporate that a single token is not enough, since users need to have backup tokens in case they lose or break these devices. As we see the threat of leaked credentials is growing and 2nd-factor authentication is providing a good solution for some, but we have to admit that solutions like hardware tokens might not scale for everyone due to their costs and management overhead.

Hence, in this work we want to show how a mobile authenticator can be built without the costs of an additional hardware device, but with similar security standards. Therefore we combine recent advancements of FIDO standards and new security features of the Android mobile operating system, to build a mobile authenticator that can be used on every FIDO compliant web service. We will explain how to implement such a mobile authenticator, how the underlying technology works and why we consider it secure. In summary, we make the following contributions:

- We discuss the issue of data breaches and how they are threatening all online services, not only those that have been attacked recently.
- We compare the current approaches for 2nd-factor authentication, discuss their limitations and identify the main reasons for the lack of adoption.
- We design and implement a mobile authenticator that combines the most recent advancements of the FIDO authentication standard and Android Security, to enable secure authentication for everyone.
- To simplify the overall process, we integrate our solution into the Avira Password Manager, which is freely available via the following link: https://www.avira.com/en/password-manager

## 2 Background

This section provides the necessary background knowledge about password security, data breaches, authentication, and security keys.

### 2.1 The Data Breach Problem

A data breach is a disclosure of private and confidential information. During the past few years, these incidents have increased tremendously. According to 'Have I Been Pwned', 11 billion user accounts from roughly 550 different websites have been compromised by data breaches so far [23]. The rising number of leaks is the result of the surging number of hacking attempts - automated phishing, malware, and brute-force attacks somewhere hit a target and allow the attacker to gain unauthorized access to databases full of user credentials [16]. When this data becomes public, it becomes a big issue for all online services, since many people reuse the same credentials on multiple services or use password patterns that are simple to guess [6]. This is how data breaches become a threat for countries, individuals, and big organizations. For example, affected companies may have to compensate their customers, become incapable of acting for weeks or months, and can even face court. Worst of all, the loss is unpredictable and can be low or high. One study from 2018 estimates that the average cost of a data breach in the U.S. is around $7.91 million, and almost 30% of all companies lose revenue after a data breach [18]. As we see, data breaches can have a huge impact, not only in a financial way but also on the operation, reputation, and image of an organization. A well-known solution to tackle this problem is to secure the logins with strong 2nd-factor authentication.

### 2.2 Hardware Authenticators

One way to integrate 2nd-factor authentication are Hardware Authenticators. By adding a 2nd layer of protection, an attacker can not log into a leaked account without the Hardware Authenticator device and the secret key it contains [19]. Hardware Authenticators are easy to use because they don't require batteries or some kind of additional software in order to run nowadays. On the other hand, a stolen or lost authenticator is an organizational disaster and can only be mitigated by adding multiple authenticators [19]. Several companies are producing Hardware Authenticators like Yubico, Kensington, and Thetis [1,25]. They are available in different price ranges, starting from $30 upwards. Security-Tokens are largely adopted by some organizations, e.g., Google, Facebook, etc., and have proven to be useful in practice [24]. The main reason for the great success of these tokens is the cooperative work of the FIDO Alliance that defines and maintains this open and independent technology for everyone.

### 2.3 FIDO

The Fast IDentity Online (FIDO) Alliance came into existence to promote new authentication standards and reduce the use of passwords [21]. Because this is an issue of many, the open industry association is supported by big companies, e.g., Amazon, Facebook, Google, Microsoft, and American Express [9]. FIDO covers a large number of technologies, including security tokens, smart cards, NFC, communication standards, and also biometrics such as fingerprint, iris,

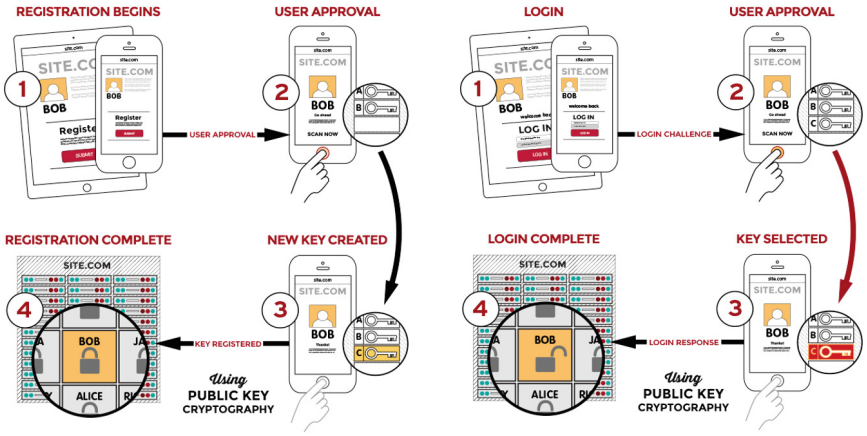


**Fig. 1.** Registration and authentication process as defined by FIDO [11]

voice, and facial recognition. The core of FIDO mainly establishes the following two processes:

i) Registration: A user receives a unique username and a randomly generated challenge. Depending on that, the user authenticator can generate a public and private key. This public key and some metadata are stored by the service [21].
ii) Authentication: The user sends the given username to the service and receives a new challenge. This challenge will be signed by the authenticator using its private key and sent back to the service afterward. The service can validate this signature using the public key of the user to verify its identity [21].

The most relevant parts of the FIDO specification for this work are FIDO UAF, U2F, and CTAP [11]. FIDO Universal 2nd-factor (U2F) specifies a universal 2nd-factor experience. The Universal Authentication Framework (UAF) defines the use of native device features like biometric authentication, e.g., fingerprint or face recognition [12]. FIDOs Client To Authenticator Protocol (CTAP) describes how the OS and a browser, can establish a connection with external devices via Bluetooth (BLE), Near Field Communication (NFC) or USB [10].

## 3 How to Solve the Data Breach Problem?

Several studies show, that breaches are getting more extensive and more frequent [6,15]. In 2019 alone, there have been at least four major data breaches, each impacting more than 200 million records. One of these, known as Collection #1, contains more than 2.7 billion email password pairs [6] and is one of the largest data breaches on the Internet. Experts have reviewed the collection and concluded that the list combined 2000 previous data breaches and added

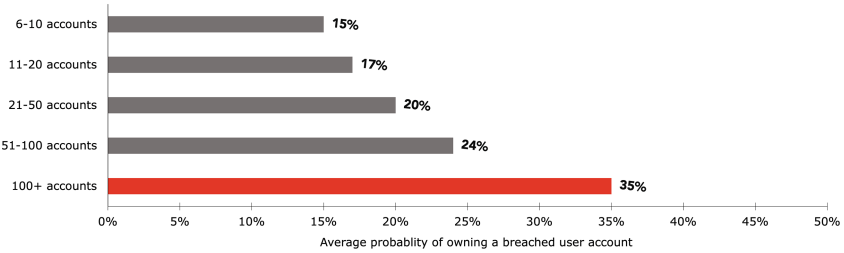


**Fig. 2.** More accounts, more breaches: The probability of owning a account that is part of a data breach increases with the number of accounts the user owns [7].

an estimate of 140 million new email addresses and 10 million new passwords from unknown sources [8]. In fact, for many breaches, it's not clear where data originates, because data get hacked, scrapped, and dropped in so many ways. Sometimes hackers are selling label the data with the name of the affected company, sometimes the data is assembled from various data breaches, and in other cases, the data is dropped without any further means. It's also not clear, why all the data get hacked in the first place because companies are often not able to detect the breaches and avoid speaking about it. But we know, there are a lot of bad security habits, such as weak and recycled passwords across various accounts, badly maintained software that can be exploited, and poorly secured databases. For example, leaked passwords are often available in plaintext rather than in their hashed version [23]. One reason might be, that passwords are not handled properly in the first place (hashing and salting passwords before they get stored in a database). Another reason is, that hackers might be able to decrypt them because weak or broken hashing algorithms have been used.

Whatever the reasons are, we often don't know, but we can measure its impact. As shown in Fig. 2, the more accounts a person owns, the higher the probability that they will be hacked. This is the result of a study conducted by Avira in early 2019 [6]. The data tell us, that users with 6 to 10 accounts have a 15% chance of a breach. This probability jumps up to 35% when the number of accounts is 100+. The main reason behind this increase seems to be the heavy reuse of account names and passwords across various online services. To follow up on this, Avira conducted an online survey with 2519 respondents aged between 20–65 years in the US [6]. A key insight from this study is, that users are more interested in simplifying authentication rather than just securing it. When they have been asked for reasons to adopt password managers, 48% of the participants said that they would adopt password managers, if they can log in more quickly and easily. A few less (44% of the participants) have indicated that they would use it to protect their passwords against hackers. Nevertheless, another study from 2019 by Pearman et al., tells us that most people don't want to pay for a password manager solution and prefer to use a free version. Just a few people said they might be willing to pay for this, and only if the tool was very secure and very easy to use [22].

In summary, science is telling us that we need to adopt more secure authentication, since data breaches and leaked accounts are threatening all online services and their users. While people seem to know about the issue, they are only ready to adapt if things are easier to use and will be free of charge.

## 4 A Secure Mobile Authenticator for Everyone

In this section, we explain how our mobile authenticator works and how we connected the various advancement of FIDO and Android OS Security to build a no-cost mobile authenticator for everyone.

Considering the recent advancements in phone security, a modern smartphone is in many ways as secure as a hardware authenticator. It can securely store private keys within the secure key storage [3] and the Trusted Execution Environment represents dedicated hardware, with well-defined cryptographic algorithms, offering just a limited attack surface [5]. In other ways, a smartphone is even superior to hardware tokens: A phone can be updated, is always with you and people already know how it's used. A hardware token on the other hand requires some effort to know how it works, you can lose or break it and updates are not supported or rather complicated. Moreover, many smartphones offer sophisticated algorithms for local authentication, e.g., finger or face recognition, something a cheap hardware token can not offer with the same level of security and useability. Finally, another great feature is to remotely find, lock, or erase the phone in case of losing it or when it gets stolen [13]. To the best of our knowledge, no hardware token offer such features, hence a stolen token requires a user to invalidate the keys and manually regain access to his accounts. Hence, we want to combine the most recent advancements in authentication standards and smartphone security to create a mobile authenticator that enables secure authentication for everyone. While keeping the same security level, as given with hardware tokens, we remove the main drawbacks such as ease of use, updatability, advanced local authentication, remote deletion, and most of all the additional costs, which are major reasons for the lack of adoption.

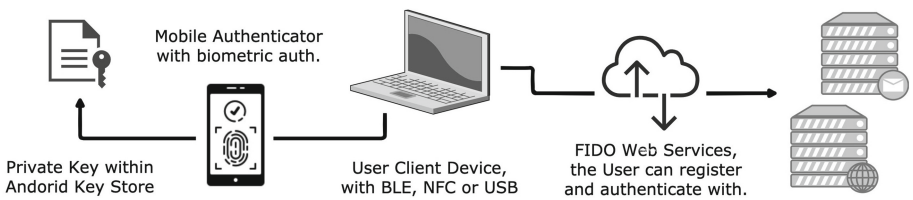


**Fig. 3.** Mobile authenticator

In Fig. 3, we show how we imagine a mobile authentication setup. While the underlying technology is much more complicated, we want to initially focus on the abstract view on the applications layer with a user's perspective in mind.

The setup comprises certain devices, owned and managed by two parties: a user owning a client device and a smartphone with the mobile authenticator installed. On the other side, a server, owned by the service that supports FIDO compliant authentication. This service can be anything from cloud services, social networks, or just a simple mail service. The owner of the service needs to implement FIDO compliant authentication. Therefore, he can use several FIDO certified third-party products, e.g. WebAuthn Awesome, that need to be integrated into the provided service [2]. While it requires some effort, it's probably one of the best ways to protect the service and its customers.

The communication between the components, shown in Fig. 3, can be established as follows. The client device has a secure HTTPS connection to communicate with the service he wants to authenticate with. In addition, another secure connection via BLE, NFC, or USB is created connection with the smartphone. The smartphone application, storing the private keys within the secure key storage, can not directly access the keys, instead, it needs to authenticate and communicate with the Trusted Execution Environment to execute cryptographic operations that use the key. When the components have been assembled in the right way, the registration and authentication procedures are ready to go.

Both registration and authentication, require 4 steps, as shown in Fig. 1. To implement these operations, we build four Java modules i) a module to establish a Bluetooth connection; ii) a module to encode, send and receive FIDO messages; iii) a module to use the cryptographic operations of the secure TEE; iv) a module to protect the access of the authenticator using biometric authentication. Using these modules we have implemented two generic functions that can perform FIDO registration and authentication procedures. Some of the implementation details can be described by stepping through the typical use cases. Please note, that we discuss the implementation for the smartphone only because the user device and web service are not part of our work.

- **Device connection:** A secure connection using BLE is created between the user device and the smartphone hosting the authenticator. For secure pairing of both devices, we had to implement an android BLE class using a GATT server. Once the devices have discovered each other, the client device gets some connection information from the authenticator and can pair the devices. When starting a 2nd-factor authentication, the browser will automatically search for a connected authenticator device, e.g., via NFC, BLE, USB.
- **Local Authentication:** The mobile authenticator app needs to be installed on the smartphone. The phone owner needs to authenticate every time he wants to use the app. The local authentication has been implemented with the face authentication procedures of the BiometricPrompt API. It supports authentication using the user's finger or iris, depending on which property is enrolled by the user.
- **Secure Key Storage:** A user can register the authenticator with any FIDO compliant service, after a successful login or even during account creation. During the registration, the authenticator needs to be unlocked and consent must be given to generate a new key pair. To generate the cryptographic keys

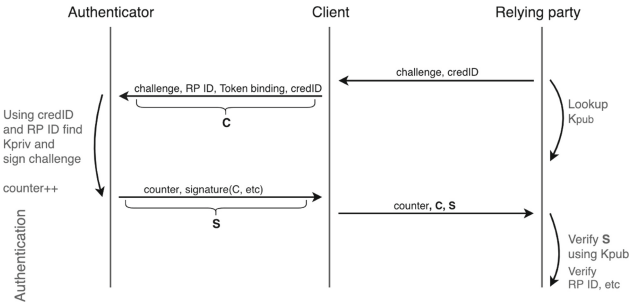


**Fig. 4.** Message flow between authenticator, client device and the service [10]

we have used the ECPublicKey class to generate a public and private key and to directly stored the private key within the Android Keystore. During registration, the public key is transferred to the service using the message API. Every registration will start another key setup and repeat the previously described steps.

– **Authentication:** After registration, the user will be asked to provide the 2nd-factor on every new login attempt. At this point, the authenticator needs to be unlocked using the local authentication feature. The authenticator sends the username and implicitly requests a challenge from the service with the message API. The message flow of this procedure is shown in Fig. 4. After receiving the challenge, the authenticator needs to look up the private key in the key manager of the Android Keystore. To sign the message within the Android Keystore we used the KeyStore signMessage() method. Afterward, the signature is passed to the app and the message API is used to transfer the signature and some additional metadata to the service. A final response will indicate if the access is granted or not.

### 4.1 Integration of the Authenticator into a Password Manager

Another idea we want to present is to integrate the mobile authenticator into a password manager application (PWM). A PWM is a tool that can create, store, and enter passwords for you in a secure way. It will store the passwords within a cryptographically secured file, that can only be accessed by entering a master password to protect the data from unauthorized access. Nowadays, not only password but all kinds of data can be stored, such as credit card information, notes, images, etc. Most vendors further provide browser extensions, smartphone apps, and online backup features to make it very convenient to use. Some popular examples are Enpass, Avira Password Manager, Bitwarden, Authy, LastPass, and 1Password.

We think, that the integration of the mobile authenticator into a PWM makes sense for two reasons: First, the user might already use the PWM to access the credentials for the first authentication step. Second, we can reduce the number

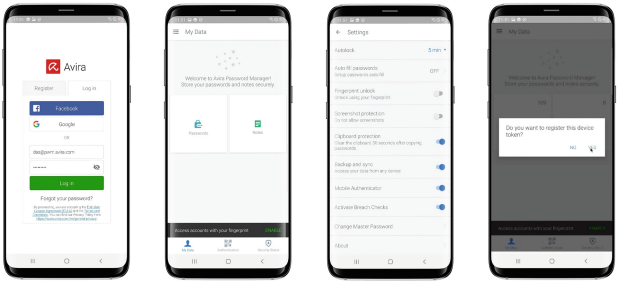


**Fig. 5.** Integration of the mobile authenticator into the avira password manager

of apps used which improves user confidence and makes it more quick and easy to use. Only a single app needs to be installed and managed to grant access to your credentials and to unlock the 2nd-factor capabilities. Furthermore, only a single authentication, using finger or face recognition, is required to unlock all the secrets and access the mobile authentication functionality.

For these reasons, we integrated the mobile authenticator into the Avira Password Manager, as shown in Fig. 5. The mobile authenticator can be enabled within the settings of the PWM. Afterward, the authenticator is active in the background and will communicate with the user via different prompts.

To implement the mobile authenticator we used Java and Kotlin, the standard programming language for Android development. Furthermore, we have used the following libraries to build our solution. Jackson data-bind library is used for data binding, which is used to convert JSON to and from plain java objects. For the UI integration of the Authenticator, we used the Android Material Design library, which provides some easy-to-use front-end widgets. Google guava API providing an advanced Java collection framework and offers a lot of handy features for functional programming, range objects, and hashing.

## 5 Discussion

In this section, we want to have a short discussion on the security of the implemented authenticator. Furthermore, we want to discuss how the authenticator compares to one-time passwords (OTP), which are often used to implement 2nd-factor authentication while avoiding the various drawbacks of hardware tokens.

### 5.1 Authenticator Security

Considering the application of our mobile Authenticator, we have to discuss its security. Our solution mainly relies on the following three security features of the Android platform security: secure key storage, strong and secure cryptographic algorithms, and a secure generation of cryptographic keys. The Android Keystore system lets you store cryptographic keys in an isolated subcomponent,

called the Trusted Execution Environment (TEE), to make it very difficult to extract key material from the device [3]. The key material is never exposed outside the TEE. If the Android OS is compromised, e.g., an attacker can read the device's memory, the attacker may be able to use any keys on the device, but can not extract it from the device. Hence, once the keys are in the Keystore, they are secure and can only be used with dedicated cryptographic operations. In addition, this operation is restricted to authenticated users only, which requires local authentication of the device owner. The Hardware security module contains only well-known and largely tested cryptographic algorithms, that are considered secure and state-of-the-art [4]. It also provides a dedicated true random-number generator to generate cryptographic keys with sufficient entropy. Furthermore, mechanisms such as resist package tampering and countermeasures against unauthorized side-loading of apps are in place to mitigate various memory attacks [5]. In summary, the Android OS includes very advanced features to provide a high level of security to protect the user's data and the mobile authenticator. Since the key will never leave the TEE, a lot of security measures are in place to prevent the key extraction. Outside of this secure environment, the communication between the devices will be secured with secure BLE pairing and HTTPS. Beyond that, when a data breach will affect one of the registered services, it can not leak any new user credentials, because only a public key and a random username is stored there.

### 5.2 Comparison with OTP

One-time passwords are 2nd-factor solutions, that are based on a shared secret and a hash function to generate new and unique passwords [17]. Comparable to the mobile authenticator, this solution requires registering a dedicated hardware or software solution with the service that can securely store the shared secret. Using this shared secret a derived password can be calculated, which can be used to authenticate to the service [17]. In the past, a lot of companies have implemented OTP, to add a 2nd-factor without the downsides of hardware tokens.

When comparing both solutions, OPT and the authenticator, are very similar but each of them has some advantages and disadvantages. Both systems rely on the availability of the device. While OTP does work even without the Internet, the authenticator requires Internet and a local connection via BLE, NFC, or USB. On the other hand, this makes the authenticator easier to use, because OTP typically requires to enter the 2nd-factor by hand and does not exchange the authentication information automatically [17]. While OTP can be synchronized with various apps like Google Authenticator, etc., the implemented authenticator is device-specific and requires a Trusted Execution Environment and advanced biometric authentication features.

Both solutions are very secure, but we think there are some major drawbacks for OTP. An attacker breaching a large authentication database will be able to generate valid OTP values at his will. With our authenticator, only a public key will be leaked, which does not threaten the user at all. On the other hand,

a lost or broken OTP token can be easily replaced with just another one that only needs to be synchronized with a service [14]. A lost authenticator instead, will be a real disaster. Once the authenticator is lost or broken, the keys can never be extracted or recovered. When changing the smartphone device, it is necessary to deregister or deactivate the authenticator for each service on its own. A centralized deregistration of all authenticator tokens managed by the app could be implemented within a PWM solution. This could help a user to disable all 2nd-factor tokens managed by the current phone before deactivating or reselling it. When activating a new phone, the user can start again to register a new authenticator with the services, without the issue of being logged out.

## 6 Conclusion

In the following, we will summarize our work and discuss the milestones we have achieved. The goal of our work was to investigate how the most recent advancements in FIDO specifications and smartphone security can be leveraged to build a secure mobile authenticator on the smartphone. To motivate our work, we discussed the various issues related to data breaches and presented some insights and statistics that have been collected from Avira's password manager. The data shows clearly that 2nd-factor authentication is a strong requirement nowadays. Hardware tokens are one way to implement this and they can prevent credential stuffing and brute-force attacks that can be affiliated to the rising number of leaked credentials and data beaches [23]. While hardware tokens are a great solution for many, they face some major issues when talking about worldwide adoption. Hence, we build a mobile authenticator that connects the most recent advancements in authentication standards and smartphone security, to enable secure 2nd-factor authentication without additional hardware cost. We discussed how to implement such a solution within the Android Trusted Execution Environment and how to integrate it within the Avira Password Manager to make the user experience more seamless and reduce user interaction. For the evaluation, we have reviewed our solution in terms of security and we compared it with one-time passwords, which are often used to implement 2nd-factor authentication without additional hardware costs. Based on our work, we perceive the mobile authenticator to be a robust, secure, and easy-to-use replacement for hardware authenticators, which can reduce the key disadvantages of hardware tokens, namely costs, management overhead, and usability.

**Acknowledgements.** The authors want to thank the Review Committee for the valuable feedback and comments. The project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 952684. Opinions, views, and conclusions are those of the authors and do not reflect the views of anyone else.

## References

1. Etienne, S.: The best hardware security keys for two-factor authentication (2019). https://www.theverge.com/2019/2/22/18235173/

2. Yuriy, A., various GitHub contributors.: Webauthn awesome. a repository with awesome webauthn/fido2 resources (2020). https://github.com/herrjemand/awesome-webauthn
3. Android Open Source Project: Android keystore system (2020). https://developer.android.com/training/articles/keystore
4. Android Open Source Project: Trusty tee (2020). https://developer.android.com/guide/topics/security/cryptography
5. Android Open Source Project: Trusty tee. androids trusted execution environment (2020). https://source.android.com/security/trusty
6. Avira Operations GmbH & Co. KG: Avira Password Security Report. Tidy up your digital life. (2019). https://www.avira.com/files/press/2019/PasswordSecurityReport_EN.pdf
7. Avira Operations GmbH & Co. KG: Avira Privacy Report. Tidy up your digital life. (2019). https://www.avira.com/files/press/2019/AviraSummerPrivacyReport_Final.pdf
8. Brian Barrett for wired.com: Hack brief: An astonishing 773 million records exposed in monster breach (2020). https://www.wired.com/story/collection-one-breach-email-accounts-passwords/
9. Octopus, C.: Fido 101: Understanding fido strong authentication and what it can do for you (2018). https://blog.strongkey.com/blog/fido-101-strong-authentication
10. Fast IDentity Online (FIDO) Alliance: Fido client to authenticator protocol (CTAP) specification (2018). https://fidoalliance.org/specs/fido-v2.0-id-20180227/fido-client-to-authenticator-protocol-v2.0-id-20180227.html
11. Fast IDentity Online (FIDO) Alliance: How fido works (2020). https://fidoalliance.org/how-fido-works/
12. Fast IDentity Online (FIDO) Alliance: Specifications overview (2020). https://fidoalliance.org/specifications/
13. Google: Find, lock, or erase a lost android device (2020). https://support.google.com/accounts/answer/6160491?hl=en
14. Grimes, R.A.: Hacking Multifactor Authentication. Wiley, Hoboken (2020)
15. Johnson, J.: Annual number of data breaches and exposed records in the US from 2005 to 2020 (2021). https://www.statista.com/statistics/273550/data-breaches-recorded-in-the-united-states-by-number-of-breaches
16. De Groot, J.: The history of data breaches (2019). https://digitalguardian.com/blog/history-data-breaches
17. Liu, Z.: An improved one-time password authentication scheme. In: 2013 15th IEEE International Conference on Communication Technology, pp. 1–5 (2013)
18. Davis, M.: 4 damaging after-effects of a data breach (2019). https://www.cybintsolutions.com/4-damaging-after-effects-of-a-data-breach/
19. Tillman, M.: What are security keys, how do they work, and which is the best to buy? (2020). https://www.pocket-lint.com/gadgets/news/150395-best-hardware-security-keys-for-two-factor-authentication
20. Mozilla Foundation : Archive of all breaches in Firefox Monitor (2020). https://monitor.firefox.com/breaches
21. Newhouse, W., Johnson, B., Kinling, S., Kuruvilla, J., Mulugeta, B., Sandlin, K.: Multifactor authentication for e-commerce: Risk-based, fido universal second factor implementations for purchasers. Technical Report, National Institute of Standards and Technology (2019)
22. Pearman, S., Zhang, S.A., Bauer, L., Christin, N., Cranor, L.F.: Why people (don't) use password managers effectively. In: Fifteenth Symposium on Usable Privacy and Security ({SOUPS} 2019), pp. 319–338 (2019)

23. Hunt, T.: Have I been pwned? (2018). https://haveibeenpwned.com
24. Yubico: Google defends against account takeovers and reduces it costs (2018). https://resources.yubico.com/53ZDUYE6/as/q3unyy-dmr8u0-fds0yi/Google_Case_Study.pdf
25. Yubico: Yubikey 5. for businesses, professionals and individuals (2020). https://www.yubico.com/de/store/#for-professionals