Privacy enhancing technologies for identity and access management

By

Michal Kepkowski

A thesis submitted to Macquarie University
for the degree of Doctor of Philosophy
Faculty of Science and Engineering
School of Computing
December 2023

Examiner's Copy

Typeset in LaTeX 2_ε.

This thesis is submitted to Macquarie University in fulfilment of the requirement for the Degree of Doctor of Philosophy.

The work presented in this thesis is, to the best of my knowledge and belief, original except as acknowledged in the text. I hereby declare that I have not submitted this material, either in full or in part, for a degree at this or any other institution.

Michal Kepkowski

Acknowledgements

First and foremost, I would like to express my deep gratitude to my thesis supervisor, Prof. Dali Kaafar, for his invaluable guidance and unwavering support throughout the course of my research. His expertise, encouragement, and constructive feedback have been instrumental in shaping the direction of this thesis and have contributed to my academic growth. I extend my gratitude to Dr. Ian Wood, whose contributions greatly enhanced my academic progress. His diligent efforts in assisting with the writing process, constructive discussions, and introduction of novel concepts enriched the content and quality of this research.

I would like to extend my thanks to Dr. Maciej Machulak for his invaluable support and contribution to this thesis. His practical insights, real-world expertise, and willingness to share industry knowledge have greatly enhanced my research, bridging the gap between theory and practical applications. His collaborative efforts and continuous feedback have been instrumental in enriching this thesis and shaping my professional as well as personal development. I would also like to express my appreciation to Dr. Lucjan Hanzlik for his indispensable help, specifically in the realm of cryptography. His deep understanding of the subject matter and insightful discussions have significantly contributed to the theoretical framework of this study.

Most importantly, I want to express my heartfelt thanks to my wife, Natalia, for her unwavering support and encouragement throughout the thesis writing process. Her belief in me, constant motivation, patience, and understanding were invaluable during challenging times, allowing me to focus on my research.

List of Publications

- JC. Polley, I. Politis, C. Xenakis, A. Master, M. Kepkowski. *On an innovative architecture for digital immunity passports and vaccination certificates*, 2021.
- M. Kepkowski*, L. Hanzlik, I. Wood, M. A. Kaafar, *How Not to Handle Keys: Timing Attacks on FIDO Authenticator Privacy* published in the 22nd Privacy Enhancing Technologies Symposium, Sydney, 2022, July.
- W. Yeoh*, M. Kepkowski*, G. Heide, M. A. Kaafar, L. Hanzlik, *Fast IDentity Online with Anonymous Credentials (FIDO-AC)* published in the 32nd USENIX Security Symposium (USENIX 2023), Anaheim, 2023, August.
- M. Kepkowski*, M. Machulak, I. Wood, M. A. Kaafar. 2023. *Challenges with Passwordless FIDO2 in an Enterprise Setting: A Usability Study*, published in the IEEE Secure Development 2023 Conference, Atlanta, October, 2023.
- I. Wood, M. Kepkowski, L. Zinatullin, T. Darnley, M. A. Kaafar. *An analysis of scam baiting calls: Identifying and extracting scam stages and scripts*, under submission, 2023.

*Principal author(s)

Abstract

The exponential growth of digital technologies has significantly impacted various aspects of modern life. This progress has particularly sparked concerns regarding the erosion of individual privacy. Notably, over recent years, there has been a substantial increase in collective awareness concerning privacy-related issues. The proliferation of social media platforms, instances of data breaches, and the commercialization of personal information have collectively fueled a heightened interest in safeguarding privacy rights. This upsurge in privacy consciousness has catalyzed a fundamental shift in the attitudes and behaviors of individuals, institutions, and governments alike.

An essential underpinning of contemporary information technology that plays a pivotal role in shaping privacy considerations is Identity and Access Management (IAM). IAM systems have emerged as indispensable tools for ensuring the security and integrity of digital identities and resources. As these systems accumulate data and facilitate the seamless access and administration of sensitive information, they inherently give rise to profound concerns regarding individual privacy and data protection. These concerns stem from the increase of data breaches and their repercussions, such as the rampant issue of identity theft, which poses substantial challenges for both organizations and governments.

In this thesis, our objective is to present a novel solution for modern IAM (Identity and Access Management) systems that enhance end-users' privacy through advanced privacy-enhancing technologies. At the same time, we aim for our solution to deliver

usability for both end-users and integrators while staying aligned with the latest advancements in the IAM industry. Therefore, as the foundation of our system, we have selected the FIDO2 protocol, an industry-recognized and widely supported solution for privacy-preserving passwordless authentication. Throughout our study, we evaluate the guarantees provided by FIDO2 to ensure its suitability for our proposed system and demonstrate how it can be used to enhance the privacy features of existing IAM systems.

We begin our study with a comprehensive exploration of usability considerations in the integration of FIDO2. While authentication is often assessed within the scope of a single application (e.g., a web application), our research delves into the challenges of FIDO2 integration across various use cases typically encountered in large organizations. We have identified both technical challenges (e.g., remote access) and non-technical challenges (e.g., lack of guidelines), taking into account a range of technologies, personas, and requirements mandated by cybersecurity and legal frameworks. Furthermore, we conducted a user study involving professionals engaged in FIDO2 integration. Drawing from over 100 responses, we organized and categorized the challenges, uncovering preferences and obstacles frequently encountered when planning or integrating FIDO2 into the existing IAM infrastructure. We contribute these findings to the FIDO2 community while also integrating them into our privacy-preserving IAM system.

Our investigation into the suitability of FIDO2 for our design progresses as we delve into an in-depth examination of FIDO2's privacy mechanisms. While the theoretical privacy assurances (such as unlinkability) align seamlessly with our requirements, we found a significant issue across major FIDO2 client implementations, which could potentially undermine the use of FIDO2 as a foundational protocol for our system. We investigated and reported a novel side-channel attack that capitalizes on a vulnerability present in major web browsers, thereby permitting remote execution of our attack. Our research identified potential adversaries and substantiated that the unlinkability property could be compromised for vulnerable FIDO2 authenticators. To address these concerns, we proposed and advocated for mitigation strategies, collaborating closely

with vendors to strengthen FIDO2 implementations. This collaborative effort ensures that publicly accessible FIDO2 clients, namely web browsers, conform to our desired requirements. In recognition of our contributions to safeguarding the privacy of users who work with Chromium-based browsers (e.g., Chrome, Edge, Opera), the Chromium security team honored us with a Chromium bounty award.

The culmination of our usability and privacy investigations has resulted in the design and development of an industry-ready and privacy-preserving system called FIDO-AC. This system facilitates the evaluation of authorization policies minimizing the exposure of private data acquired from trusted sources (e.g., ePassports). Notably, our proposed solution binds advanced privacy-enhancing technologies with FIDO2. This fusion enhances privacy-preserving authorization in tandem with robust authentication, forging a cohesive link between these processes. Our design effectively tackles usability challenges through its seamless and adaptable integration with existing FIDO2 deployments. We regard FIDO-AC as a comprehensive solution tailored to both industry and academia. Therefore, our contribution encompasses a theoretical definition of the framework, a formal analysis of security and privacy, a detailed design of an exemplary instantiation of the system, a proof-of-concept implementation demonstrating integration with Android OS and ICAO ePassports, and an assessment of the implementation's viability.

Through the contributions presented in this thesis, we aim to promote the adoption of privacy-enhancing technologies within IAM systems. By identifying practical challenges and vulnerabilities and subsequently introducing an innovative solution to address them, we provide a well-founded approach toward privacy-conscious IAM systems. Notably, we anticipate that our FIDO-AC system will considerably enhance the privacy attributes of the existing authentication and authorization methods.

Contents

List of Figures

List of Tables

1

Introduction

In the increasingly interconnected world, the protection of individuals' personal information and the consequent preservation of their autonomy and security are of utmost importance to society. Unfortunately, an individual's privacy can be infringed upon in numerous ways. In the context of contemporary communication technologies, the most prominent form of privacy violation pertains to the improper handling of Personally Identifiable Information (PII), encompassing its processing, storage, and sharing.

Before the advent of the Internet revolution, controlling PII was relatively straightforward due to limited avenues for information transportation. However, this scenario shifted with the escalating reliance on digital data records and the rapid exchange of information, primarily propelled by technological advancements and the widespread use of the Internet. The rapid pace of globalization and the diminishing barriers between systems have ushered in greater challenges in safeguarding private information. This

problem is amplified by prevailing practices of collecting more data than strictly necessary for operational purposes. Whether driven by the advantages of retaining data for analytical insights, legal obligations, or the ease of system implementation, the principle of data minimization has often been disregarded. The upshot of this state of affairs is that even a minor security breach can result in a significant compromise of private information. Recent instances of data breaches, such as the 2022 Optus case [29] and the 2023 Latitude hack [30], underscore the inadequacy of security measures employed by companies that handle PII. Despite the emergence of new industry solutions and government regulations (e.g., mandates for adopting multi-factor authentication), the incidence of such breaches continues to rise [1].

The security of data collection and storage relies heavily on the quality of access controls. However, even when the access management (AM) layer is correctly configured, the multitude of actors, roles, and use cases can complicate control efforts. For instance, an application granted privileged access to the data may inadvertently create a pathway for data leakage beyond control. Interestingly, Personally Identifiable Information data isn't solely imperiled by external threats. As highlighted in 2019 Verizon's report [188], insider actors frequently target medical and personal data. Regrettably, in most instances, enforcing restricted access control becomes challenging due to the roles assumed by these actors. Organizations often opt to implement Data Loss Prevention (DLP) systems to mitigate insider threats. However, these systems only address a limited range of use cases, such as monitoring email channels.

Particularly susceptible to data leaks are applications operating under the business-to-consumer model, where users serve as the ultimate recipients of services. Collection of PII data is typically initiated during the early stages of interactions with applications, with the aim of establishing identities (i.e., digital user representations) and managing access. Identity and Access Management (IAM) systems serve as foundational components for introducing user-centric controls, collectively referred to as IAAA (Identification, Authentication, Authorization, and Accountability). For instance, the acquisition of a copy of an identification document, such as a driver's license, might be necessary to conduct identity verification procedures. Notably, authentication and

authorization hinge on user data to confer access to resources and services, making them the primary focus of this thesis.

Authentication is a process employed to verify identity ownership, in which the identity can be represented by an opaque and privacy-preserving identifier. However, for usability reasons, industry practice often involves the use of easily recognizable identifiers, such as email addresses. Among the most prevalent authentication solutions, password-based authentication holds a prominent position. Notably, within the realm of privacy, the *something you know* authentication factor (e.g., passwords) offers robust privacy preservation. However, due to the multitude of potential attack vectors, relying solely on this factor is no longer recommended [153]. A widely embraced solution involves augmenting the authentication process with a second authentication factor, known as two-factor authentication (2FA), which usually relies on the *something you have* factor. This combination substantially increases resistance to common attack methods (e.g., password spraying [160]). Nonetheless, maintaining privacy can be challenging within this framework. For instance, a widely adopted approach involving out-of-band authentication via one-time passwords sent over a telephony network necessitates a phone number, which can consequently serve as a unique identifier for the user.

Modern authentication methods, incorporating Multifactor Authentication (MFA), have been gradually gaining traction. For instance, Microsoft reported a 7% increase in accounts with MFA in 2022 [133]. Among the MFA implementation choices, the *something you have* factor (e.g., SMS one-time passwords) has gained significant popularity. However, recent data breaches (such as those affecting Uber [195], Dropbox [116], and Cisco [115]) have revealed that vulnerabilities persist in certain MFA methods. These vulnerabilities include issues like MFA fatigue [117] and Adversary-in-the-Middle (AitM) attacks [18].

In response to emerging attack vectors on authentication, identity experts led by the FIDO Alliance have proposed a potential solution known as FIDO2 [39]. The FIDO2 protocol was designed to offer local verification (e.g., using fingerprints or PIN)

and passwordless cryptographic authentication, effectively constituting an MFA solution on its own. FIDO2's robust security and privacy framework, coupled with its widespread adoption, positioned it as a paradigmatic modern authentication protocol. Consequently, this thesis will focus its contributions on conducting an in-depth examination of the FIDO2 protocol.

An important aspect of the IAAA principles as they relate to privacy is authorization, which closely follows authentication and involves making decisions based on rules (also known as policies) regarding whether an authenticated identity can gain access to requested resources or services. Authorization systems typically strive to strike a balance between control complexity and granularity. Consequently, role-based access control (RBAC) is a favored choice for commercial solutions. Roles serve as a grouping mechanism, simplifying the assignment of privileges (e.g., an admin role having *write* access to all resources). Nonetheless, this straightforward model can lead to scalability issues (such as role explosion) and challenges in integration with dynamic environments.

Particularly within contemporary security architectures reliant on a continuous flow of signals, authorization frameworks need to evaluate a multitude of parameters in a detailed manner (e.g., by ingesting signals from the Continuous Access Evaluation Protocol). This scenario provides an ideal use case for attribute-based access control (ABAC) systems. These systems collect signals and user attributes, forwarding them to a central location for policy assessment. Interestingly, constructing meaningful policies requires a comprehensive set of user attributes. For instance, a commonly encountered scenario involves an age restriction policy hinging on the "date of birth" attribute. Similarly, a policy that enforces location-specific rules requires the collection of geolocation data. As demonstrated by the examples above, more precise authorization policies demand more accurate user attributes, unfortunately introducing a privacy impact.

The privacy considerations highlighted above for authentication and authorization underscore the fact that privacy is often not the primary focus when designing identity and access management processes. Regrettably, the absence of the *privacy by design*

principle in IAM systems leads to the creation of PII data repositories, thus increasing the risk of unauthorized data leaks. Notably, solutions aimed at constructing private, secure, and finely-tuned access control systems do exist, but they are frequently underutilized in practice. Despite extensive research into privacy-enhancing technologies, their adoption within the industry is marginal and often misaligned with commercial objectives. As a result, it becomes imperative not only to propose novel designs of privacy-enhancing solutions but also to implement them as default options within industry solutions. This could involve incorporating such solutions as integral components of commercial products. A perfect illustration of such integration is Brave Web Browser [40], which provides a Tor network client by default, increasing usability and availability of advanced privacy protection.

In this thesis, we examine the usability of the integration (Chapter 3) and privacy mechanisms of FIDO2 (Chapter 4). This study involves a comprehensive examination of requirements, persistent challenges, and prevalent security and privacy issues within the existing FIDO2 deployments. These identified areas serve as a foundational basis for the subsequent design, development and implementation of an innovative privacy preservation system, described in Chapter 5 and referred to as FIDO-AC. The primary focus of the FIDO-AC system is to effectively mitigate privacy-related concerns inherent in the domains of authentication and authorization, while seamlessly interfacing with pre-existing industry deployments, thereby addressing usability concerns.

The initial part of our research provides a comprehensive overview of the FIDO2 protocol and its privacy-preserving features as well as an overview of emerging privacy-preserving technologies in the IAM industry. In Chapter 3, we conduct a usability study to identify the real-world challenges faced by professionals when adopting robust and privacy-preserving authentication methods. While the security, privacy, and usability of FIDO2 have been extensively discussed in the academic and industry literature, the complexities tied to its integration into production environments, including solution completeness and edge-case support, have remained relatively understudied. Notably, environments characterized by intricacies like customer and enterprise identity and access management present distinct challenges for any authentication system.

Through our usability study, involving insights from over 100 professionals rooted in their hands-on field experience, we pinpointed demanding identity lifecycle use cases (e.g., remote access and legacy systems). Importantly, our findings enabled us to categorize and prioritize these challenges, including technological problems such as server integration as well as non-technological obstacles such as insufficient level of knowledge. Our contributions are not limited to the analysis and exploration. We extend actionable guidance to the FIDO community for future advancements, underscoring the significance of usability — a principle that was our priority while designing our privacy-enhancing framework for FIDO2 in Chapter 5.

In Chapter 4, we delve deeply into the privacy-preserving aspects of FIDO2. Interestingly, our research uncovers a novel side-channel attack capable of compromising the unlinkability of vulnerable FIDO2 implementations in the wild, including hardware authenticators and popular web browsers. We demonstrate that in vulnerable authenticators, there exists a time disparity between processing a key handle for a distinct service on the same authenticator and processing it for a different authenticator but the same service. This timing discrepancy can be exploited to execute a timing attack, permitting an adversary to establish links between users' accounts across various services. We offer multiple real-world instances where adversaries positioned to execute our attack can capitalize on linking accounts. Remarkably, our testing revealed that two out of the eight hardware authenticators assessed were susceptible, even though they held FIDO level 1 certification [68] (a certification program led by FIDO Alliance to evaluate security and functionality of FIDO2 authenticators). Our attack shows how privacy can be easily compromised, despite the commendable security controls upheld by the FIDO community and major technology vendors. The significance of our finding has been recognized by the FIDO Alliance, authenticator vendors, and major browser vendors. In particular, our contribution to identifying and mitigating privacy issues in Chromium-based browsers (e.g., Chrome, Edge, Opera) has been recognized by the Chromium security team and honored with the Chromium bounty award.

The privacy and usability investigations have led us to surprising conclusions. Notably, we observed that even when employing FIDO2 privacy enhancements during

authentication, there exists limited to negligible privacy protection during the authorization phase. Furthermore, the FIDO2 protocol lacks specifications to integrate trusted attributes with the FIDO authentication process in a comprehensive manner, enabling users to selectively disclose them to the relying party as needed. Essentially, applications requiring authorization (e.g., age or driver's license expiry date verification) still rely on ad-hoc methods that do not satisfy the data minimization principle and deny users the ability to verify the data they disclose.

To address the aforementioned issues and the usability challenges identified in Chapter 3, we introduce a FIDO-AC system. In Chapter 5, we present a detailed proposal for this novel system, which combines the FIDO2 authentication process with the user's digital and non-shareable identity. Significantly, FIDO-AC integrates privacy-enhancing technology to ensure that the verifier solely gains knowledge of the authorization policy output. We achieve this feature by utilizing zero-knowledge proof (ZKP) technology [198], which makes it possible to prove a statement is true while preserving the confidentiality of secret information (PII in our case). Additionally, ZKP provides anonymity and unlinkability of the data holder, perfectly aligning with the FIDO-AC system requirements. Apart from the primary goal (i.e., maximizing user's privacy), we designed FIDO-AC as a framework with the following properties in mind. First, the process of sharing credentials has to implement active authentication (also known as a liveliness check) to ensure that the user is in possession of non-shareable attributes (e.g., ePassport). Simultaneously, we enforce our design to follow the user-centric approach and impose minimal friction on the existing FIDO2 flow. This applies to both end-users and implementers; therefore, we ensure that FIDO-AC is fully compatible with FIDO2 deployments, and the functionality extensions are of a pluggable nature (i.e., no need to modify existing applications). Finally, we defined an efficiency constraint that mandates reasonable performance and scaling capabilities (when compared to regular FIDO2).

Our contributions extend beyond the system's design; we also provide an extensive security and privacy analysis, along with a complete implementation of the system. We present the process of instantiating our framework using off-the-shelf FIDO2 tokens and

any electronic identity document, such as the ICAO biometric passport (ePassport). Importantly, we showcase the system's usability by seamlessly integrating it with existing FIDO2 clients and authenticators. Finally, we demonstrate the feasibility of our approach by evaluating a prototype implementation of the FIDO-AC system.

2

Background and Literature Review

In this chapter, we discuss the state of authentication in the wild and introduce FIDO2, a protocol we carefully examine in Chapters 3 and 4, and use as a building block in Chapter 5. We provide an overview of its processes (registration and authentication) and explain supporting mechanisms such as user verification. We discuss the security and privacy considerations, as well as the implications of various types of FIDO2 authenticators. Please note that the abbreviations FIDO and FIDO2 are used interchangeably, as they refer to the same protocol.

We supplement the background of our research with an overview of privacy methods in identity and access management systems. We present the latest advancements from regulatory bodies, as well as community initiatives. We list the PETs technologies that have the potential to be used or are already integrated into IAM solutions. We conclude this chapter with a literature review which focuses on the systems that extend or build

upon the FIDO2 protocol as well as systems that implement privacy protections into the authorization processes.

2.1 Authentication Landscape

Authentication methods have undergone substantial development in reaction to the emergence of novel attack vectors, resulting in considerable diversity in the landscape of authentication methods in practical use. Organizations are faced with the challenge of adapting to the rapidly evolving authentication landscape, taking into account factors such as acceptable risk, security profile, and financial resources, among others. However, the process of selecting an authentication approach (e.g., one-factor vs. multi-factor), determining additional procedures (e.g., re-authentication or step-up authentication), and, ultimately, deciding on factor type and implementation, often presents obstacles and can lead to delays in the adoption of modern authentication methods.

Insights into the decision-making processes employed by organizations to select their authentication methods are documented in the NIST 800-63B publication [153]. This document comprehensively addresses authenticator types, various implementation strategies, and methodologies for selecting authentication mechanisms. The specification lists three authentication factors, namely, *something you know*, *something you have*, and *something you are*. It not only provides illustrative examples but also explores potential combinations to achieve the desired security level. A detailed examination of the selection framework reveals that a high risk assessment within any evaluation category (e.g., financial loss, reputation damage, criminal violations) leads to the adoption of the most secure authenticator category. This typically entails the use of multi-factor cryptographic devices, such as hardware tokens secured with local fingerprint verification, thus advocating strong authentication for the majority of deployments.

While the NIST 800-63B document offers valuable insights into authentication trends, it falls short of providing a comprehensive depiction of the prevailing state

of authentication in real-world scenarios. Unfortunately, conducting an exhaustive and global assessment of existing authentication deployments presents cannot be done directly, as it necessitates a valid user account to trigger the authentication process. Nonetheless, partial glimpses into trends within deployed systems can be gleaned from industry reports and surveys. For instance, a 2022 survey of authentication trends for eIDAS (European eID regulation) published by Sharif et al. [171], indicates that the most commonly adopted authentication method (utilized in 52% of member states) involves the combination of a password with a second factor, in which 16% rely on SMS one-time passwords. Authentication methods achieving a high assurance level, as defined by ISO 29115 [106], such as software authenticators, smart cards, and secure keys, are implemented in 33% of member states.

An alternative perspective is provided by the FIDO Alliance through their 2022 FIDO Authentication Trends report [73], which presents findings from a user study examining authentication methods employed for accessing financial services. Respondents disclosed that passwords remain the most prevalent method (51%), followed by biometrics and one-time passwords (at 30% and 28%, respectively). Physical security keys were utilized by a mere 9% of respondents

2.1.1 Attack Vectors

Authentication systems, as direct gateways to digital systems, are continuously exposed to variety of malicious activities. Below, we describe a spectrum of attacks and the gradual improvements in the IAM industry to secure digital systems.

The oldest digital authentication method, based on memorized secrets (i.e., password-based authentication), is vulnerable to a spectrum of attack vectors. The static and user-friendly nature of passwords renders them vulnerable to various forms of guessing attacks. The adversary's toolkit includes techniques such as naive brute force enumeration [148], dictionary-based enumeration [22], password spraying [163], as well as more targeted approaches [191]. Notably, adversaries do not always need to resort to blind password guessing. Numerous data breaches [101], alongside other credential theft methods [182] (e.g., keyloggers), furnish them with a substantial repository of

credentials that can be repurposed in credential stuffing attacks [137].

The widespread adoption of a second factor of authentication, often relying on generated codes, has notably increased the complexity of attacks. Out-of-band methods, such as SMS one-time passwords or push notifications, have effectively mitigated guessing attack vectors. However, this has compelled adversaries to alter their tactics, shifting towards attacks centered around channel manipulation, such as replay attacks [143], downgrade attacks [184], SIM swapping [112], MFA fatigue [117], and adversary-in-the-middle [18] configurations.

Another consequence of fortifying authentication measures is the shift in adversaries' focus toward persuading users to grant them access to the system. Unfortunately, adversaries have correctly identified that the human factor is often the weakest link in authentication schemes, requiring less effort to exploit than highly secure authentication methods. Consequently, attacks targeting users, such as phishing [142], pharming [141], and social engineering [94], have witnessed a significant surge [1; 94] since the widespread adoption of two-factor authentication. According to Grimes et al. [9], as much as 90% of data breaches in the comprehensive privacyrights.org dataset are attributed to attacks capitalizing on human vulnerabilities.

Notably, adversaries combine technical and human-oriented attacks to formulate strategies targeting modern authentication systems. Recent security breaches at companies like Uber [195], Dropbox [116], and Cisco [115] underscore the fallacy of relying solely on the secure behavior of human actors. Phishing, often coupled with adversary-in-the-middle attacks, or MFA fatigue in conjunction with social engineering, have introduced attack vectors that present challenges for both password-based and out-of-band MFA methods. In response, the IAM industry has shifted its focus toward cryptography-based approaches designed to address the aforementioned issues, including those originating from human error. The subsequent section delves into one such cryptography-based solution that has gained popularity in contemporary IAM deployments.

2.2 FIDO2

FIDO2[1] is an authentication protocol designed by FIDO Alliance in collaboration with vendors and identity and access management (IAM) experts. The open nature (i.e., free and publicly available specifications)[2] and broad support (i.e., all major web browsers and operating systems) make FIDO2 a serious candidate for becoming the de facto standard for second-factor and passwordless authentication.

The FIDO2 standard [71; 189] defines two processes (ceremonies): registration and assertion. The former allows the creation of a link between the server and the authenticator. The latter is used to prove the authenticator's possession (e.g., a cryptographic token). Both are built on a simple request-response transaction that generates verifiable proof. Usually, three parties participate in the flow: FIDO Server, FIDO Client, and the authenticator. As presented in Figure 2.1, the authentication flow starts with a trigger sent to the FIDO Server (steps 1. and 2.). The trigger might be an automated action or user interaction (e.g., the user clicks the login button). Then, the FIDO Server generates a random challenge that travels through the FIDO Client to the authenticator (steps 3. and 4.). In WebAuthn, the user's action is required to unlock the authenticator (step 5.). Finally, the authenticator generates a signature (using the preregistered key) and sends it back to the FIDO Server (steps 6. and 7.). The transportation layer of FIDO2 is composed of two related protocols: CTAP[71] and WebAuthn[189]. The former is responsible for communication with an authenticator (i.e., binary messages sent via USB, BLE, or NFC channels), whereas the latter describes the API for the client side.

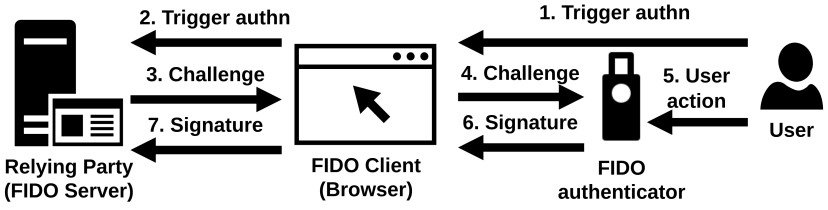


FIGURE 2.1: FIDO2 parties and simplified authentication flow

[1]https://fidoalliance.org/what-is-fido/
[2]https://fidoalliance.org/specifications/

FIDO2 can be a second factor or a single multi-factor authentication method. Multi-factor property is usually achieved by implementing a local user verification (e.g., fingerprint scan) to unlock the private key, and thus combining *"something you are"* with *"something you have"* factors. It is worth mentioning that the passwordless configuration is not the only flow. FIDO2 supports a username-less flow, in which the user identifier is not revealed to the FIDO server (i.e., the selection of the account is done locally), increasing the privacy and usability of authentication.

2.2.1 FIDO2 Ceremonies

In the FIDO2 specification, we distinguish two phases: *registration* and *authentication*. We will now describe this process for a typical use case where the client is a browser, the authenticator is a USB hardware token (implementing industry-grade signature schemes such as ECDSA - Elliptic Curve Digital Signature Algorithm [111] or RSA[165]) and the relying party is a standard web-server.

Registration

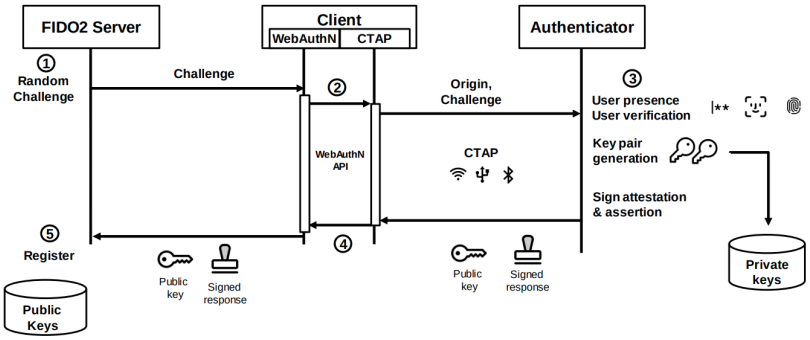


FIGURE 2.2: FIDO2 simplified registration flow.

The purpose of registration phase (presented in Figure 2.2) is to bind the authenticator to the user's account. Similar to the standard login/password based scenario the user is provided an interface to send out registration requests to the server. Prior to receiving such a request the server generates a unique challenge value (step 1.) and

returns it to the browser which handles the whole registration process using a server-provided JavaScript based application. The browser then uses credential management API which internally executes the CTAP protocol with the authenticator (Step 2.). The challenge and additional data like the server origin in the form of an application ID are sent to the authenticator using one of the channels (i.e., USB, BLE, or NFC).

After the user presence check, the authenticator internally generates a key pair for the signature scheme and uses the secret key to create the assertion to the server's challenge (step 3.). The client's request specifies whether the secret key should reside on-device or the key pair is of the non-resident type. In the latter case, additionally to the response, the authenticator also returns a key handle (a random identifier of the key). Depending on the implementation of the non-residents key this can be either a random value that is used as input to a KDF (key derivation function) or the encryption of the secret key for ECDSA.

Finally, the public key, key handle, assertion, and optional attestation (of the public key) are sent to the server (step 4.). The data is then verified and if accepted the public key and key handle are added to the database (step 5.).

Authentication

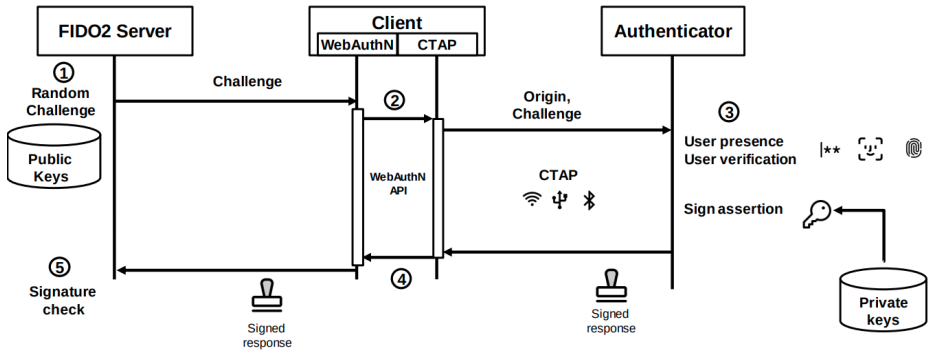


FIGURE 2.3: FIDO2 simplified authentication flow.

After successful registration, the user now tries to access her account and is prompted to enter credentials. This action starts the authentication process (presented in Figure 2.3) which generates a random challenge on the server side (step 1.). Additionally, it

allows the server to locate the user's account and the devices bound to it. Note that it is commonly allowed to register multiple devices with a single server. A list (also called allowed list) of key handles corresponding to those devices is sent to the client which tries to find the right key handle. This is internally done by asking the hardware token to authenticate giving each element on the list as input (step 2.). Note that if those key handles correspond to different devices there will be only one key handle that will generate a valid assertion. Interestingly enough this internal verification is performed *without* user presence check since otherwise the user would be required to click a button multiple times (depending on the size of the allowed list). After the right key handle is found the browser issues the last call to the token and requires a user presence check (step 3.). Finally, the authenticator creates the assertion which is a digital signature on a message containing, among others, the server's challenge, an authentication counter (to protect against cloned devices), and the origin. The browser sends the assertion to the server (step 4.) that provides access to the service if the verification was successful (step 5.).

2.2.2 Passwordless Authentication

FIDO2 represents a revolutionary advancement in the realm of customer IAM, enforcing an authentication paradigm shift. Despite past attempts to replace weak authentication methods, such as Windows CardSpace [98], no other technology has achieved a comparable level of adoption as FIDO2. This widespread acceptance can be attributed to several key factors, including native support from operating systems, cross-vendor endorsement from leading companies, and its open standard nature. However, the primary advantage of FIDO2's extensive adoption lies in its ability to offer ubiquitous and seamless passwordless authentication.

From a security and privacy perspective, FIDO2, similarly to other cryptography-based authentication methods, relies on digital signatures transmitted from authenticators to servers. This mechanism effectively eliminates the need for users to use weak credentials (e.g., passwords) for authentication. While weak credentials like passwords

or PINs may still exist, their purpose is confined to local verification, primarily unlocking the authenticator itself. Consequently, user credentials are never transmitted to the remote server, significantly reducing the scope for potential attacks on both security and privacy.

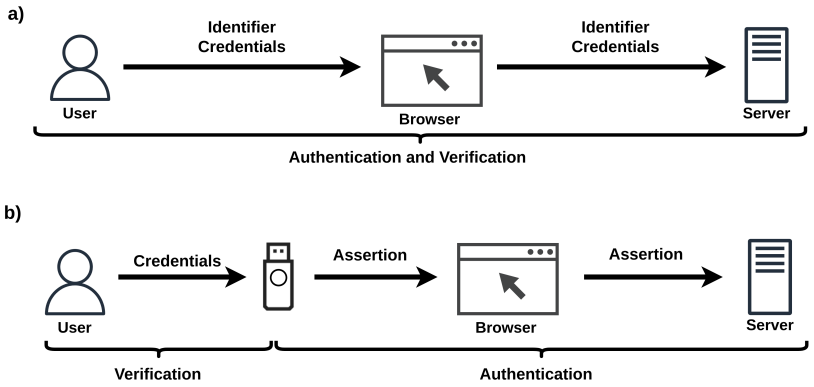


FIGURE 2.4: Conventional vs FIDO2 passwordless authentication.
a) Conventional authentication flow using credentials such as passwords.
b) FIDO2 passwordless authentication flow.

In Figure 2.4, we illustrate the conceptual contrast between conventional authentication methods (e.g., using username and password) and the passwordless approach implemented by FIDO2. In the former case (a), data sent over the internet includes an identifier and credential susceptible to a wide spectrum of attacks. In contrast, the latter case (b) involves transmitting only an assertion, i.e., a signature over a random challenge. This mechanism employed by FIDO2 offers significant mitigation against modern types of attacks, such as phishing with an adversary in the middle.

2.2.3 User Presence and Verification

The core mechanism of FIDO2 authentication embodies the *something you have* authentication factor. However, to increase the security level, the majority of FIDO2 implementations incorporate a human-in-the-loop requirement. It is noteworthy that bypassing this requirement is feasible, a process referred to as silent authentication. Nevertheless, silent authentication is disabled in the WebAuthn API [189], limiting its application to customized implementations. During the authentication process, the

user may be prompted to prove their presence by pressing a button on the authenticator. This action unlocks the authenticator and generates an assertion.

For scenarios demanding heightened security measures, such as protection against physical token theft, FIDO2 authenticators can integrate two authentication factors: *something you have* (i.e., the authenticator) and either *something you know* or *something you are* (e.g., fingerprint). In such cases, the device requires the user to verify the transaction. For instance, the user might be required to provide their PIN or fingerprint to unlock the authenticator, allowing for authentication to proceed.

Importantly, the type of user check (i.e., user presence or user verification) and the type of verification are included in the assertion, allowing the server to validate them. Leveraging the trust established through the authenticator attestation, the server can implement authorization policies accordingly.

2.2.4 FIDO2 Security and Privacy

FIDO2 is designed to leverage the cryptographical authentication scheme (that is, the verification of the signature over a challenge) to achieve passwordless authentication. The formal security model and analysis of WebAuthn and CTAP2 were studied by Barbosa et al. [31] and later extended with formal privacy analysis by Hanzlik et al. [99]. Theoretical academic analysis together with the FIDO Alliance FIDO Security Reference [70] give a solid foundation for the FIDO protocol and position it as a strong candidate to increase the security of authentication globally.

The security properties of the FIDO2 protocol address popular attack vectors in existing second-factor deployments (described in Section 2.1.1). First, FIDO2 saves the registration's origin together with the private key, which is then verified in the authentication phase. In the web context, the origin is based on the web page's domain. This simple yet effective technique allows for mitigating a popular attack scheme that lures users to authenticate on the malicious domain (i.e., phishing with Adversary-in-the-Middle). The attack typically starts with a phishing message that contains a redirection link to the malicious server. The server can replicate the login page or proxy the traffic, becoming Adversary-in-the-middle (AiTM). In the case of FIDO2

authentication, the login attempt will fail because FIDO2 is used directly in the channel for establishing a session, and thus the registered origin will not match the one provided during the authentication.

The second class of attacks, which is easily mitigated with FIDO, is the MFA fatigue class. The attack focuses on popular out-of-band authentication methods (e.g., push notifications) and targets the human factor of authentication. The mechanism of out-of-band authentication requires a trigger action (e.g., sending SMS OTP), which can be abused by adversaries to generate a malicious MFA prompt (e.g., if the first-factor credentials are known). The next stage of the attack involves social engineering to convince users to accept the authentication request. Unfortunately, as discussed in Section 2.1.1, relying on the human factor to determine the authenticity of the MFA prompt is not a secure design. In the FIDO2 protocol, an MFA fatigue attack cannot be used because authentication has to be triggered and completed by the same actor through the same channel (e.g., web session).

Furthermore, FIDO2 effectively addresses well-known vulnerabilities stemming from password-based credential attacks. For example, database leaks pose no harm as FIDO2 servers only store public keys. Moreover, FIDO2's utilization of a digital signature scheme makes guessing attacks infeasible, while the integration of random challenges for each transaction serves to mitigate replay attacks. Although FIDO2 significantly enhances security, it is not entirely impervious to theft and eavesdropping. For instance, while challenging, it is not impossible for an individual to observe a PIN and subsequently steal a physical token. Nonetheless, such scenarios require a substantially higher level of effort and complexity compared to traditional password-based attacks. Additionally, FIDO2 offers inherent resistance against social engineering attacks, given its inability to be easily shared or remotely triggered.

The design of FIDO2 follows the *"privacy by design"* principle. Unlike other methods that require Personal Identifiable Information (PII) to function (e.g., phone number in SMS OTP case), FIDO2 does not need any PII. In fact, for discoverable credentials (i.e., a configuration of private keys), no user-related information, not even an identifier, is required to perform authentication. Furthermore, FIDO2 ensures that the protocol

does not compromise privacy (e.g., by linking accounts). Both registration and authentication ceremonies incorporate mechanisms to prevent privacy leakage. Firstly, for each registration, a new key pair with randomly looking key handles is generated, and thus it is not possible to link users based on the server-side data. Similarly, the authenticator attestation mechanism is designed to prevent a unique identification of the authenticator.

2.2.5 Hardware vs Software

FIDO2 authenticators can be implemented as a platform, integrated with the device, or as roaming authenticators, implemented either as a purely software solution or based on hardware elements such as Trusted Platform Modules (TPMs). The primary distinction lies in the location of the cryptographic operations, which is readily understandable; however, the implications of this decision may not be immediately apparent. Presented below are our considerations regarding the types of FIDO2 authenticators.

Firstly, we examine the construction of authenticators. Hardware authenticators execute critical operations on dedicated elements specifically chosen and integrated to fulfill a single task: the execution of the FIDO2 protocol. Consequently, they offer a heightened level of trust, aligning well with the Separation of Duties design principle. In contrast, software authenticators operate within a multifaceted environment and frequently share hardware resources with other processes. Although Trusted Execution Environment (TEE) environments presently elevate the security standards, they are unable to match the same level of security as hardware-based solutions. Notably, vulnerabilities in TEEs have been previously identified[45].

Vulnerabilities can occur in both hardware and software authenticators; however, the functionality and complexity of hardware token firmware are significantly lower compared to software authenticators, which often serve multiple purposes. Hardware authenticators adhere to the security design principle of “Keep it Simple,” reducing the risk of vulnerabilities compared to software authenticators. Additionally, the risk of exploiting vulnerabilities should be considered.

Hardware authenticators, especially roaming ones, are less likely to be exploited due

to their typical activation only during authentication and operation within a single process environment. Conversely, software authenticators are continuously active in a multi-process environment, increasing the surface and time window for potential exploitation. As a result, the risk of exploitation is smaller in the case of hardware authenticators.

The lifecycle of software and hardware authenticators differs. Hardware authenticators, particularly roaming ones, are delivered as immutable devices, while software authenticators have regular update schedules. This difference has interesting consequences when considering legislative requirements for authenticator evaluation and certification. Frequent updates pose challenges in meeting these requirements. On the other hand, discovering a vulnerability in a hardware token, although rare, may necessitate a recall of all devices, posing a significant risk of business operation interruption.

Although FIDO2 adheres to the open design principle, where security relies on well-established cryptographic primitives, the level of protection against reverse engineering within implementations can significantly contribute to threat mitigation in production deployments. Hardware authenticators employ measures such as limited APIs and closed structures, rendering them challenging to compromise. In our research, we discovered only one documented instance of such a compromise in the public literature[14]. This case necessitated extensive professional expertise and the use of expensive specialized equipment, presenting a substantial barrier to malicious operators. Conversely, software solutions are comparatively easy and inexpensive to inspect, even if they incorporate countermeasures such as obfuscation, and have experienced multiple compromises in recent years[45]. This is a fundamental reason why hardware tokens are considered to offer better guarantees in terms of protection against reverse engineering and side-channel attacks.

2.3 Privacy-Enhancing Technologies

Privacy-Enhancing Technologies (PETs) offer a range of innovative solutions aimed at enhancing privacy properties of digital systems. PETs encompass mathematical mechanisms designed to strengthen privacy attributes such as anonymity or unlinkability. In this section, we discuss privacy goals and the technologies that achieve those goals for IAM use cases. Subsequently, we explore which technologies have been integrated into the IAM industry.

2.3.1 Privacy Goals

In this section, we introduce privacy goals as outlined by Pfitzmann et al. [156], specifically: anonymity, pseudonymity, unlinkability, undetectability, and unobservability.

Anonymity is typically defined within the context of a subject that generates observable events, such as communication between a user and IAM system. It asserts that an adversary should be unable to identify the event's creator from the list of potential subjects (i.e., anonymity set), thereby preserving the true subject's anonymity. The requirement for anonymity frequently arises in systems where adverse repercussions can be imposed upon subjects for their actions, as exemplified in whistleblower systems.

Anonymity brings significant benefits to subjects' privacy; however, it also makes processes such as auditing, troubleshooting, or revocation difficult to achieve. A more relaxed approach to anonymity, known as pseudonymity, can be used to strike a balance between privacy and usability. The goal of pseudonymity is to conceal the identities of subjects (e.g., the digital representation of a person). Pseudonyms can take the form of random numbers, email addresses, or cryptographic certificates, effectively substituting the actual identity of the subject with usually unrelated values. Typically, issuers of pseudonyms (e.g., IAM systems) maintain a link to the subject's true identity, which means that the pseudonymization process can be reverted, for example, to audit the subject's actions. However, for parties without the link (e.g., adversaries), pseudonyms effectively hide the true identity.

Subject identification is not the only relationship of interest to potential adversaries. Notably, knowledge of the connections between events can compromise the subject's privacy. Specifically, associating the same subject with multiple events disrupts the unlinkability property. From an attacker's standpoint, unlinkability indicates that the adversary is unable to adequately differentiate whether events that occur within a system are interconnected or independent. Unlinkability is often regarded as a crucial attribute for systems in which the tracking of users' activities has the potential for privacy and security loss (e.g., payment systems).

Linking events to the subject necessitates adversaries to possess the capability to ascertain the existence of a message. However, certain systems demand an even more robust property known as undetectability, which ensures that attackers are incapable of adequately distinguishing whether the event indeed exists or not. Furthermore, one might also require anonymity for the subjects implicated in the event. This combination of properties, denoted as unobservability, although challenging to realize in practice, embodies the most stringent privacy objective that we discuss.

2.3.2 IAM Related Technologies

The aforementioned privacy goals pose distinctive challenges for system implementers. Typically, production systems, including IAM systems, primarily employ rudimentary techniques such as data obfuscation, encryption, or federated identity to mitigate the risk of private data exposure. However, these fundamental approaches fail to address all scenarios in which private data is needlessly collected or shared. Especially within IAM systems, processes related to identity (e.g., authentication, authorization) necessitate a more sophisticated approach to ensure privacy. Below, we discuss advanced PETs and instances from the literature that exemplify advancements in privacy within the realm of IAM.

Sharing private data, such as users' identities, is a significant concern for IAM systems. In particular, use cases that necessitate data exchange between IAM systems pose a risk to data privacy. In some instances, PETs can be applied to mitigate

these risks. For example, a limited set of queries between systems (e.g., finding common users) can be conducted in a confidential manner using Private Set Intersection (PSI) [77]. PSI allows two or more mutually distrustful parties, each with their own input sets (S1 and S2), to confidentially determine their shared elements, represented by the intersection $S1 \cap S2$. Notably, the intersection is achieved without revealing any non-intersecting elements to the other party. An illustrative example of PSI's application in IAM is presented by Liu et al. [128] in their "SEPSI" protocol for the Internet of Things (IoT), where identity information is embedded in a private manner. Another noteworthy application of PSI is found in the work of Yuanhao et al. [192] in their "APSI" system, which not only combines PSI with authorization but also supports flexible authorization and cross-type authorized comparisons of data sets.

IAM systems contain a significant amount of private data. Analysis of this data, if performed honestly, can benefit data owners (e.g., by providing discounts to some users). However, if a malicious actor gains access to a data set intended for analysis, it can result in a significant privacy breach. This problem can be mitigated through the use of Differential Privacy (DP) [61]. This method involves adding noise to algorithmic outcomes, with the aim of minimizing the disclosure of sensitive information related to the inputs used in the computation. The amount of noise introduced is carefully calibrated to provide mathematical assurances regarding input privacy. By employing this approach, individuals' private data remains protected while enabling meaningful analysis and computations. An illustrative application of DP is presented by Yao et al. [196] in their differential privacy-preserving user linkage framework for accounts in online social networks. This framework enables cross-network capabilities, such as analytics, while safeguarding individual privacy.

Storing and processing private information represents one of the core functions of IAM systems. Particularly, remote biometric authentication (i.e., executed on the server) raises concerns regarding the security and privacy of the biometric features stored within IAM systems. This concern arises due to their immutable nature (e.g., fingerprint features are unique and do not change), making them impossible to reset if stolen. One solution to mitigate the risks associated with potential biometric data

leaks is Fully Homomorphic Encryption (FHE). This concept, introduced by Rivest et al. [164], involves a unique form of encryption. With FHE, individuals possessing a designated public key have the ability to encrypt information, while only the owner of the private key can decrypt it at a later stage. Furthermore, by leveraging the encrypted data and the public key, anyone can perform computations on the encrypted information and evaluate algorithms that utilize confidential inputs. In the context of biometric authentication, conducting calculations in an encrypted domain provides significant enhancements in security and privacy for the biometric data stored in IAM systems. An illustrative example of this approach is presented by Vallabhadas et al. [186], who utilized FHE for iris and fingerprint features to substantially improve the security and privacy of biometric data.

One of the drawbacks of typical IAM systems is their centralized nature, often controlled by a single entity, such as a company or government. Processes like pseudonymization or authorization policy evaluation can be compromised if the centralized party becomes untrustworthy. Some of these issues can be addressed through the use of Secure Multiparty Computation (MPC) [85]. MPC protocols enable collaboration and computation of a shared function using private inputs from multiple parties. MPC ensures that only the resulting output of the function is disclosed, while all other details about the inputs remain concealed. The resilience of MPC against participants who intentionally deviate from the protocol's guidelines can be enhanced, and generally, the security and privacy of MPC are maintained as long as a specified threshold of participants does not collude to compromise them. In the context of IAM, an example was presented by Sucasas et al.[176], who proposed an approach to construct an MPC-based pseudonym system for smart cities without the need for a trusted issuer. Another example in the healthcare sector was provided by Tan et al.[178], who proposed a pre-authorization system based on MPC to enable advanced consent for accessing medical records.

Transactions such as payment orders or vote submissions typically require authentication. IAM systems often implement re-authentication or step-up authentication to verify transactions and attribute the issuer's identity through digital signatures.

However, there are situations where revealing identity information in transaction authentication is not desirable, as in the case of anonymous eVoting. In such cases, the privacy of transaction authentication can be achieved through the use of blind signatures [48]. An extension of digital signatures, blind signatures enhance privacy by allowing a user to obtain a signature from a signer on a document without the signer having access to the actual content of the "blinded" document being signed. Consequently, if the signer is later presented with the signed "unblinded" document, they cannot associate it with the signing session or the individual on whose behalf they signed the document. An example of the application of blind signatures in an eVoting system was presented by Kumar et al. [122]. Their system ensures vote privacy in an end-to-end verifiable manner.

Authorization policies are typically assessed using data, such as location, which can be considered private. Given that IAM systems cannot always be considered trustworthy, sharing private information for authorization purposes can pose potential privacy risks. To mitigate this concern, Zero-Knowledge Proofs (ZKPs) [86] can be employed. ZKPs are cryptographic algorithms that enable a prover to convince a skeptical verifier of the truthfulness of a specific statement. Although the verifier possesses knowledge of the statement, often represented as a program, the proof itself (e.g., an input causing the program to output 0) remains undisclosed to the verifier. An example of the application of ZKPs for location data authorization is presented by Jagwani et al.[109], illustrating how ZKP technology can be applied to a system to prevent location tracking.

The aforementioned methods provide strong privacy properties; however, they are still in the innovation phase in the technology adoption curve. Complexity, awareness, expertise, stable implementations, and the absence of standards are just a few factors that deter industries from upgrading their IAM systems to incorporate privacy-preserving capabilities. Notably, the provision of a highly private environment may sometimes clash with business strategy (e.g., when the collection of PII is integral to a business model), thereby constraining the adoption of advanced PETs. Therefore, in the next section, we explore what privacy-preserving technologies are actually used in

production deployments.

2.3.3 Privacy Preservation in IAM Industry

Within the domain of IAM, in particular Customer IAM, organizations strive to strike a delicate balance between providing secure and convenient resource access while concurrently upholding privacy rights and mitigating the collection and exposure of personal information. Unfortunately, despite the advancement of PETs, conventional IAM systems tend not to adopt privacy-enhancing technologies, thereby engendering privacy risks and potential data misuse. The 2023 OECD report "Emerging privacy-enhancing technologies" [145] states that PETs are usually not being integrated into the production IAM systems due to outdated legislation, knowledge gaps, and technological difficulties. As stated by the OpenID Foundation in its 2023 report on the Privacy Landscape of Government-Issued Digital Credentials [147], only basic privacy-enhancing methods such as data collection minimization, are used for public digital identity systems.

The IAM community notably strives to provide privacy-oriented solutions for commonly used protocols. For instance, proposals like "Privacy Enhancing Mobile Credentials" from the Kantara Initiative advocate for the adoption of privacy-preserving solutions. Similarly, privacy-focused efforts have been undertaken within standardization bodies. For instance, ISO/IEC 24745:2022 on Biometric Information Protection [108] defines privacy protection requirements for biometric data. Additionally, the Selective Disclosure for JWTs (SD-JWT)[67] specification illustrates how JSON Web Tokens (JWT)[105] can be issued in a manner that allows the token holder to apply selective disclosure when presenting the token, releasing only specific claims as needed. Although based on the simple principle of signing hashed claims, SD-JWT can be readily applied to the majority of identity protocols, such as OAuth 2.0 [100] or OpenID Connect [129]. Furthermore, industry is actively working on standardizing cryptographically advanced methods. In particular, decentralized identity and verified credentials work groups[3] are exploring privacy-preserving methods for sharing Personally Identifiable Information

[3]https://www.w3.org/2022/06/verifiable-credentials-wg-charter.html

(PII) and have introduced the BBS Signature Scheme [130] specification. This scheme enables implementers to achieve selective disclosure, unlinkable proofs, and proofs of possession in a standardized manner using verifiable credentials.

The application of PETs in identity and access management often encounters limitations stemming from usability, legal requirements, and business considerations. Simple customer IAM use cases, such as federation and single sign-on, as well as more complex use cases in financial institutions, like know your customer and anti-money laundering, pose challenges for enhancing privacy with PETs. Specifically, arguments against increased privacy often cite concerns regarding accountability and revocability. Nevertheless, as listed by Baum et al.[33], several solutions providing advanced PETs have been successfully implemented in production. Notable examples include Microsoft's U-Prove[151], IBM's Idemix[42], and the Hyperledger Foundation's Indy project [2].

2.4 Literature Review

In our pursuit of solutions for crafting a privacy-preserving system for IAM, we undertook an exhaustive review of the existing literature. Our goal was to identify systems that encompass, to some extent, our desired features which include smooth integration with established FIDO2 deployments, robust privacy preservation mechanisms, and authorization grounded in verifiable attributes. Presented below are our findings, which encompass systems utilizing FIDO2 as a foundational component (without protocol adjustments), systems that adapt or enhance FIDO2 to build novel functionalities, and systems that prioritize the integration of robust privacy measures into authentication methods widely used in the industry.

The growing popularity of FIDO2 has captured substantial attention within the academic community, resulting in numerous proposals for constructing systems grounded in FIDO2. For instance, the "Let's Authenticate" system introduced by Conners et al. [53] combines FIDO2 with privacy-preserving certificates rooted in a well-established PKI model. Notably, FIDO2 functions as a robust authentication mechanism to a

Certificate Authority (CA), which subsequently transitions to certificate-based authentication. An intriguing application of FIDO2-based systems arises from research centered on the identity aspects of smart grids. Farao et al. [64] unveiled a system called P4G2Go, delivering a privacy-preserving framework for roaming energy consumers, such as tenants in rental apartments. This scheme hinges on passwordless authentication built upon FIDO2 and incorporates privacy technologies through the Idemix implementation. Further examples emerge from the European Union, where FIDO2 integration finds its place within eIDAS-enabled systems. For instance, the self-sovereign identity system put forth by Bolgouras et al.[37] presents a unified design incorporating the latest advancements in the IAM sector. In a similar vein, the Incognito project[150] introduces a comprehensive system harnessing FIDO2, verified credentials, and privacy-preserving sharing capabilities. While these systems leverage FIDO2, they lack a protocol-level integration of FIDO2 with attributes sourced from verifiable origins, making them inadequate for constructing our envisioned system.

Another avenue of FIDO2-related research centers on circumventing constraints inherent in the specification and publicly accessible FIDO2 implementations, often achieved by introducing custom extensions to the protocol. For instance, Wagner et al.[190] examined authentication for devices with constrained interfaces (such as smart TVs) and introduced a proxy-based system. This system facilitates secure FIDO2 transactions between limited FIDO2 clients and FIDO2 authenticators. Notably, Hackenjos et al.[97] presented an intriguing extension to FIDO2 functionality. Their FIDO2D framework is tailored to enhance the security of critical web transactions. It achieves this by implementing one-out-of-two security, ensuring resilience even if one of two devices is compromised, and by introducing transaction authentication. The authors achieved this by modifying the FIDO2 transaction to bind two consecutive FIDO2 authentications from separate devices. In a different vein, the concept of continuous authentication has been explored within the FIDOnuous system by Klieme et al.[114]. Their goal is to thwart malicious actors from hijacking authenticated sessions by intermittently conducting silent FIDO2 authentications throughout the session's duration. An innovative application of the FIDO2 protocol was introduced by Whalen

et al.[194] who proposed a FIDO2-based human verification system. This system aims to supplant commonly used but less user-friendly methods (such as CAPTCHAs) with FIDO2 attestation. Utilizing the information encapsulated in the FIDO2 attestation, including user presence proof, and trusting the security of authenticators via vendor-issued attestation certificates, this system employs FIDO2 registration to verify the physical presence of a human actor. An inspiring piece of work was presented by Schwarz et al. [170], who introduced a system called FeIDo. This system effectively tackles the FIDO2 recovery challenges through the utilization of dynamically generated FIDO2 credentials. The authors make use of verified attributes from eID to successfully regenerate FIDO2 credentials. This noteworthy work has motivated us to integrate eID data into our own system. Similarly, the system presented by Okawa et al. [146] has provided us with insights into the fusion of attributes with FIDO2 for evaluating authorization policies. Their design capitalizes on an attribute-based signature framework to assess authorization policies based on FIDO2 signatures.

The aforementioned examples offer valuable insights into how FIDO2 can be harnessed to create systems with heightened security and privacy attributes. Unfortunately, all the showcased systems make substantial modifications to the FIDO2 protocol, resulting in compatibility issues with real-world deployments. Extension models built upon core FIDO2 protocol adjustments do not align with our objectives and therefore cannot be integrated into our system.

The concluding segment of our review encompasses privacy solutions tailored for industry-ready authentication and authorization systems, which also hold a prominent position as research topics within academia. For instance, Roy et al. [168] propose an extension to a widely used method for accessing remote servers, specifically Secure Shell (SSH). The authors employ a variant of broadcast encryption known as anonymous multi-KEM to embed privacy features into SSH. This authentication mechanism thwarts privacy-targeted attacks such as client de-anonymization or user probing. Another noteworthy system, named PrivateDrop, is introduced by Heinrich et al. [102] to enhance the security of Apple's AirDrop functionality. This system introduces a

privacy-preserving approach, leveraging private set intersection (PSI) technology during the contact discovery phase of AirDrop, enabling mutual authentication of transactions. A distinct PETs technology, namely revocable keyed-verification anonymous credentials, has been applied to real-life use cases by Dzurenda et al. [63] within their Privacy-Enhancing Authentication System (PEAS). This system provides a means to anonymously access both electronic services (e.g., web services) and physically protected areas (e.g., parking lots, offices). Notably, PEAS offers privacy-preserving guarantees that align with the objectives we seek for our system. However, it is constructed upon a proprietary protocol that isn't easily compatible with existing IAM deployments, rendering it unsuitable for our specific use case.

The examined systems present intriguing FIDO2 extensions and advancements in authentication privacy. While they may not align with our desired features, they provided us with an inspiration for shaping our own approach.

3

FIDO2 Integration Usability Challenges

This chapter is adapted from the work titled "Challenges with Passwordless FIDO2 in an Enterprise Setting: A Usability Study" published in IEEE Secure Development 2023. Authors: M. Kepkowski, M. Machulak, I. Wood, M. A. Kaafar*

Every emerging technology, including FIDO2, inherently conforms to the technology adoption life cycle [134]. Commencing at the innovation phase, if proven successful, the technology progressively transitions into the sphere of "early adopters," eventually permeating the "early majority" segment. However, the trajectory of this transition and its widespread integration within user communities and industries is far from a trivial endeavor. Such an evolution frequently reveals a plethora of usability challenges, originating from both end users and integrators.

In particular, the incorporation of cybersecurity solutions demands a substantial effort to ensure optimal functionality. A notable illustration of this is the study by

Krombholz et al. [120] of pertinent issues in the TLS configuration. In the context of our research, which aims to introduce privacy-preserving technologies into IAM systems, it is imperative to underscore the considerable usability risks to both security and business operations. Since IAM processes directly impacts the user experience and overall perception of the service, a comprehensive understanding of the prevalent usability challenges becomes an essential prerequisite to the design of any privacy-enhancing extensions.

In this chapter, we explore the usability challenges with the identity lifecycle within the FIDO2 protocol. Notably, we focus on understanding these challenges from the perspective of integrators, encompassing software engineers, development and operations (DevOps) specialists, and managers. To further strengthen our analysis and substantiate our findings, a comprehensive user study was conducted, in which information was obtained from cybersecurity professionals, delineating their perspectives on issues associated with the integration of FIDO2. Importantly, the selected findings of this chapter constitute the foundational principles underlying the design criteria of the FIDO-AC framework, as presented in Chapter 5.

3.1 Introduction

Online authentication is critical for the security posture of every data-driven organization and is one of the main pillars of an emerging security design paradigm called zero trust [166]. Additionally, increasing customers' and workforce awareness of their data privacy forces organizations to introduce new privacy-preserving methods for authentication and authorization. Perhaps at the expense of usability, organizations have been slowly rolling out additional authentication factors (e.g., in 2022, Microsoft reported 7% increase of accounts with MFA [133]). A *something you have* factor (e.g., SMS one-time passwords) is currently a popular choice for the MFA implementation. However, as discussed in Section 2.1.1, the popular MFA methods continue to exhibit vulnerability to modern attack vectors, thereby posing a risk of private data leaks.

A potential solution to address modern attack vectors on authentication and safeguard privacy, called FIDO2, has been proposed by identity experts led by the FIDO Alliance (see Chapter 2 for the FIDO2 protocol description). According to digital identity hype cycle modelling by Gartner[17], FIDO2 is expected to become a dominant solution for strong authentication in the next 2-5 years. For this to happen, FIDO2 needs to be widely adopted by the industry. However, compared to popular MFA methods, FIDO2 is a complex protocol and its security and privacy are largely dependent on its reliable implementation, deployment, and maintenance by all parties (i.e., authenticator, client, and server). New technologies, including FIDO2, have to overcome adaptation challenges before reaching a critical mass. While the FIDO2 security and privacy properties as well as end-user usability are well studied [31; 99; 83; 149], protocol adaptation in complex environments is rarely discussed, even though technology uncertainty as defined by Stock et al. (e.g., complexity) has a major impact on integration success [175].

In this chapter, we explore how FIDO2 as a passwordless solution and its deployability are perceived in complex settings. We analyze and discuss the views and experiences of 118 professionals involved in FIDO2 deployment and draw conclusions about challenging aspects of FIDO2 adaptability and usability, some of which can pose a serious risk to the FIDO2 popularization process. In particular, we aimed at answering the following question: What are the technological (e.g., implemented functionalities) and non-technological (e.g., know-how) challenges that discourage enterprises from integrating FIDO2-based passwordless authentication?

3.2 FIDO2 Adaptability in Diverse Environments

FIDO2 represents a paradigm shift for commonly used authentication usually based on passwords. Before FIDO2, passwordless authentication was used primarily in highly secure environments (e.g., smart cards used in government agencies and financial institutions). With FIDO2, passwordless authentication was introduced for everyday use cases, which raises questions about usability and adaptability. Regarding end user perception, academics as well as industry bodies have outlined challenges based on the usability studies. For example, Lyastani et al. [83] conducted a comparative user study on 94 participants and found out that even though FIDO2 passwordless authentication was well received, usability concerns such as recovery from the lost authenticator were present. Similarly, Owens et al. [149] examined the user's perception of passwordless authentication using RAs and reported users' concerns regarding availability, account recovery/backup, and setup difficulties. However, the success of new technology such as FIDO2 depends not only on the end-user experience but also on how adaptation of a new technology and its operation are perceived by integrators. In particular, in large organizations, these aspects are of pivotal importance in technology selection. Interestingly, this aspect of FIDO2 usability has received little attention in academia.

The Identity and Access Management (IAM) capability in large enterprises differs from IAM in smaller organizations. The disparity can be attributed to a number of factors such as the size, complexity, use cases, technology that is used, and regulatory or legal obligations, among others. Unlike smaller organizations, large enterprises typically rely on multiple authentication systems to ensure the security of their assets. Moreover, these organizations are often complex with numerous locations around the globe. As such, authentication needs to be suitable for various devices and systems, including those used to access company assets. Additionally, it must also be able to cater to a large number of users, including those with various persona types that have different and often conflicting authentication requirements.

The diversity of authentication requirements results in the growth of complexity and cost of moving from one authentication mechanism to another. For instance,

simply 'switching on' FIDO2 on a dedicated Identity Provider (IDP) is insufficient. A new mechanism is needed to address all possible authentication routes. In such cases, enterprises must consider the types of authenticators suitable for their persona types, whether platform or roaming (hardware or software), and their security properties. They must also ensure compatibility with existing or planned authentication systems such as IDP compatibility. Additionally, some use cases may require authenticators to comply with such certifications as FIPS 140-2 [144] or NIST 800-63B [153] while also providing biometric-based user verification on top of simple user presence checks.

Support and toolings are equally important to allow for full authenticator life-cycle management, including strong credential binding but also secure recovery and fallback processes. In more sophisticated cases, organizations may look into attestation to have central and policy-based control over authenticators. They may even consider baking their own, unique key into the authenticators for their workforce. In contrast, consumer-oriented systems prioritize usability and most FIDO2-compliant devices can be considered suitable for authentication.

3.3 FIDO2 Use Cases

Evaluation of security protocols is usually done in theoretical or laboratory conditions [31; 99], which can make it detached from real-world issues. In particular, authentication protocols are closely coupled with human-focused processes. For example, the JML (Joiner, Mover, Leaver) process [26] is usually used in organizations to define rules and requirements for the identity lifecycle and, by extension, for authentication-related processes. To understand the challenges identified in the usability study (Section 3.4), we provide below a brief review of the FIDO2 ecosystem in the context of identity lifecycle use cases. We present the processes in chronological order, reflecting the user's interaction with the system. In particular, we focus on processes, which were found challenging by the participants of our usability study (see Section 3.4). As illustrated in Figure 3.1, we commence with the handover, binding, and provisioning processes, which are crucial for ensuring the user's ability to authenticate successfully

and access the system. Subsequently, we delve into typical authentication use cases, such as remote server access. Following that, we examine self-management processes, including migration and account recovery. Finally, we discuss the end of the lifecycle, during which a user's account is deprovisioned. Please note that custom or edge cases are not included, however, we believe that the described use cases, defined based on the industry-focused literature review [179; 135; 124; 72], give a representative set of use cases.

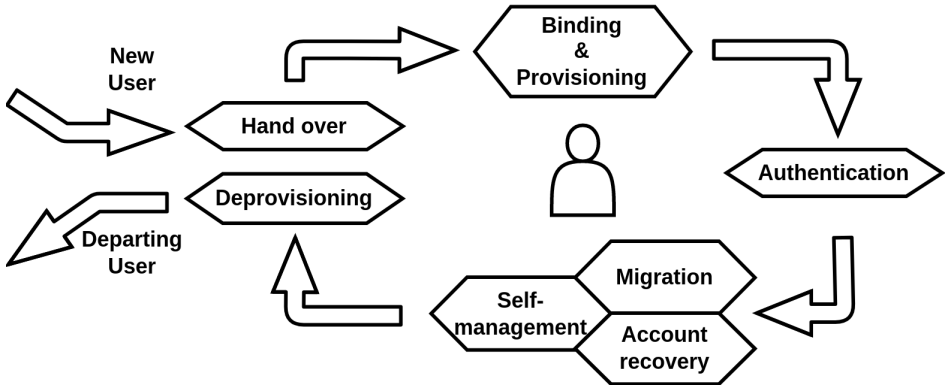


Figure 3.1: Authentication related processes in the identity lifecycle.

3.3.1 Authenticator Hand Over and Binding

The handover process is the first step to onboard a user with FIDO2 authentication. In case of hardware RA, a physical device (e.g. a USB token) needs to be provided to the user. Clearly, this operation opens a path for malicious actors to break authenticator security even before it is used. Attacks such as device cloning, firmware modifications, or key extraction in the supply chain can significantly decrease the trust for token-based passwordless schemes. Even though the mitigation of handover risks is out of the scope of FIDO2, vendors as well as the protocol itself provide tools to reduce some of these risks.

Authenticator authenticity protections increase trust that hardware RA has not been tampered with. Following the guidelines provided by Pfeffer et al. [154], software, hardware, and packaging countermeasures can be used. For software authenticity,

manufacturers can execute local or remote validation of firmware based on hash or signature. Another approach is to delay the moment of firmware load and initialization till after RA is delivered. In terms of hardware authenticity, an RA usually leverages a secure CPU (or co-processor) to perform cryptographic operations and keep the keys secure. Additionally, some vendors enhance their products with tamper-detection circuits and single-piece casts. Finally, the packaging of the delivered token can be designed to prevent tampering with the RA without visibly damaging the package or holographic sticker.

FIDO2 was designed to provide a secure method of authentication with privacy by design to satisfy not only enterprise but also public use cases. One of the privacy-preserving mechanisms, preventing tracing of an authenticator, can be found in device attestation. The same identifier and vendor certificate is shared by a significant number of devices (over 10000 [74]), thus preventing unique identification. However, this feature also makes device filtering, which is a desirable security control in an enterprise, impossible. Therefore, in the recent CTAP 2.1 version [71], an additional way to attest a device, called "Enterprise Attestation", was provided. Enterprise attestation allows FIDO-relying parties to request a uniquely identifying attestation during credential registration. The details of creating authenticators with enterprise attestation are out of the scope of FIDO2 but ideas on how to do it have already been proposed by vendors (e.g., unique identifier in the device attestation) [197].

Once a user receives an authenticator, the binding process is used to create a link between (user) identity and authenticator (a public key in the case of FIDO2) [153]. Interestingly, the protocol does not specify the process of verifying the identity. However, because the security of the scheme heavily depends on the proper identity assignment, we explore possible approaches to secure the binding of authenticators in an enterprise setting.

Firstly, an authenticator can be bound to the identity before it is delivered to the user. Such a process, either remote (e.g., RA sent to the user) or in-person is costly and difficult to scale. Alternatively, binding can be established during the first enrollment (e.g., for new employees). As described in the FIDO Alliance guidelines [69], based on

the required assurance level, one of the following models can be used. Firstly, the Trust on First Use (TOFU) model binds an authenticator to a new unknown account, the Invitation model leverages the user's pre-collected data (e.g., email address) to provide a unique binding mean (e.g., one-time use link). Additionally, a 3rd party process can be used to validate identity. For example, an external Identity Provider (Federation model) or Identity Verifier (Identity Proofing model) can certify the user's identity. Finally, authenticator binding can be required for existing users (Post-enrollment binding). In this case, an existing authentication with the equivalent security posture can be used (Anchor model). In all cases, it is pivotal that the binding is done during a strongly established session to prevent unauthorized binding attacks [153].

The diversity and complexity of potential enrollment procedures (e.g., handover, binding, and provisioning) described here pose significant difficulties in constructing and implementing a secure sequence. It is worth noting that experts involved in our study also recognized the enrollment processes as challenging (see Section 3.4). In the following sections, we delve into the identity processes subsequent to account enrollment.

3.3.2 Authentication

In this section, we discuss authentication use cases typically encountered in diverse environments. We examine approaches to remote authentication, as well as processes that alter the authentication context, such as privileged access.

Telework Authentication: We examine how authentication in remote work scenarios is being addressed by FIDO2, following the NIST SP 800-46r2 [135] categorization of remote work (telework).

Telework Client Devices such as mobile phones, personal PCs, and laptops usually implement local verification based on credentials such as PIN or biometrics. While mobile devices have already implemented FIDO2, laptops do not yet implement this protocol pervasively (at the time of writing, only the Windows Hello for Business framework provides out-of-the-box FIDO2 support), which makes a FIDO2 rollout challenging in the diverse enterprise environment.

A direct connection with the application server (Direct Application Access), even though simple, is not always an option due to security and regulatory requirements (e.g, applications are inside a tightly controlled perimeter). A common solution is to apply an additional layer of security (and authentication) between the telework client devices and servers. Usually, this is enforced using tunneling, application portals, or both. Tunneling is often implemented as a VPN (Virtual Private Network), which creates a secure connection between the client and the VPN gateway, thus allowing the client to connect to application servers. Alternatively, application portals (e.g., virtual desktop infrastructure) move the application clients into a controlled environment, through which they are accessed. In both cases, a client has to authenticate. For both mobile and web applications, FIDO2 provides a well-defined and supported authentication method. This method can be easily implemented for application portals, whereas for VPN clients, the support depends on the vendor's implementation.

Notably, enterprise authentication is not limited to applications, but to infrastructure as well. Remote access to servers is equally essential to the company's operations, regardless of technology (graphical or command-line-based). In Windows environments, Remote Desktop Protocol (RDP) [132] is commonly used, however, it does not support FIDO2 yet[1]. In the case of Linux servers, Secure Shell Protocol (SSH) [138] is a common form of remote access. At the time of writing, OpenSSH supports FIDO2 only as a way to protect private keys[2].

Credential Delegation: Typically employees' accounts are bound to a single identity (e.g., through the enrolment process). However, single ownership of an account (and associated credentials) can be insufficient to cover more sophisticated use cases - e.g. multiple individuals may need access to a shared resource and this requires credential delegation. Such delegation is often accomplished by credential sharing which makes it impossible to trace and hold individual users accountable for their actions on that resource. There are however other approaches for credential delegation. As described by Grosse et al. [91], the best delegation approach is one directly implemented in an

[1] https://learn.microsoft.com/en-us/azure/active-directory/authentication/howto-authentication-passwordless-security-key-windows

[2] https://developers.yubico.com/SSH/Securing_SSH_with_FIDO2.html

application (e.g. calendar sharing), however, this approach does not scale. Alternatively, as shown in the AARC project [20], delegation can be realized using protocols such as OAuth2.0, yet this approach requires additional infrastructure and is focused on delegating access to a non-human actor via APIs.

The FIDO2 protocol does not natively support credential delegation - tight binding of user verification (e.g., fingerprint) with the authenticator makes sharing of a FIDO2 credential impractical. However, Frymann et al. [78] describe the use of the Asynchronous Remote Key Generation (ARKG) primitive to generate credentials in FIDO2 which can be delegated.

Shared Credential: Shared credentials are a common use case in enterprises. As outlined by Haber et al. [96], some devices and applications are built with only a single local account, and thus a common practice is to share this credential (e.g., password or private key) among the team members. FIDO2 was not designed to facilitate the shared credential scenario. However, one could configure the FIDO2 authenticator to be shared by the team (e.g., a single RA without user verification). While the FIDO2 authenticator provides better security properties than passwords or private keys (e.g., anti-cloning measures), it does not solve the fundamental issues with shared credentials (e.g., lack of accountability).

Privileged Accounts: Privileged access is common in enterprise environments. Dedicated Privileged Access Management (PAM) solutions are used to supervise how (privileged) accounts are used [110]. As described by Habel et al. [95], PAM systems typically provide the functionality of credential vault, session proxy, and audit register. Following the Zero-Trust [79] properties with concepts such as least privilege model [140] and Just-In-Time provisioning [139], PAM systems serve to significantly mitigate the potential for unauthorized access, thus effectively minimizing associated security risks.

FIDO2 in the context of PAM can be considered not only as an authentication to access PAM vaults. For example, dynamically created SSH keys (for privileged accounts) can be secured with a personal FIDO2 token (ecdsa-sk key type), thus introducing user verification for each use of SSH keys. FIDO2 can also facilitate continuous authentication that could ensure the integrity of privileged sessions routed via PAM

proxies. Although continuous authentication is not directly supported in FIDO2, the primitives such as silent authentication (without human action) are already in place. An example of how this could be achieved using FIDO2 extensions was proposed by Klieme et al. [114].

Even though the authentication use cases listed above are not exhaustive, they demonstrate the variety of constraints and requirements that an authentication system needs to address. Notably, the popular use cases already pose challenges (see our user study results) and these are usually amplified by custom (i.e., business-specific) and edge case variations.

3.3.3 Authenticator Migration

In an enterprise setting, devices may have a predefined lifespan (2-3 years according to the Gartner report[13]) and may be replaced to reduce the cost of the maintenance of the old devices, which has a direct impact on authenticator management. FIDO authenticators usually store private keys in a Restricted Operating Environment, such as TEE based on ARM TrustZone hardware [27]. This environment applies Key Protection Security Measures (SM-1 [75]), which prevent key export and is considered a desirable security feature. However, this security feature also introduces a significant trade-off between security and usability. When a device is replaced, the usability of the platform authenticator (PA) is diminished, and employees are required to re-register the authenticator with the system each time a new device is issued. Alternatively, they can use RAs such as USB tokens. But even RAs can be affected by device replacement, for example, tokens with a USB-A port may be incompatible with USB-C only machines. Additionally, the process of re-registering an authenticator needs to be secured at least as well as the initial credential binding process. Interestingly, industry experts proposed a solution named multi-device credentials [76] (a.k.a. passkeys), however, the relaxed security model (e.g., extractable keys) makes them not suitable for the secure enterprise environments.

These considerations add complexity to FIDO2 deployment, and as shown in the results of our user study, authenticator migration is perceived as a challenging process.

3.3.4 Account Recovery

Account recovery is the process of regaining access if the primary authentication method cannot be used (e.g., lost or stolen authenticator). Usually implemented as fallback authentication, account recovery has to be done using authentication which is at least as secure as the primary method. Various fallback authentication methods, including their usability and security properties, have been evaluated by academics. AlHusain et al. [24] provided an extensive review of fallback authentication research and concluded that the most popular methods are based on mobile devices. An assessment of the usability of fallback authentication was done by Markert et al. [157]. Their preliminary study shows that SMS and email-based methods are more usable than other approaches.

Account recovery procedures for the workforce notably differ from those designed for the customer experiences. As outlined by Saxe et al. [169], enterprise processes focus on security and access continuity. Additionally, the reduced user base (i.e., only personnel) and usability requirements, allow leveraging human-based solutions (e.g., help desk or peer checks). Even though such methods are not flawless (e.g., prone to social engineering), their flexibility and non-programmatic nature contribute to access continuity. Interestingly, as shown by Reynolds et al. [161], over 16% of help desk tickets in their study were related to account recovery.

The security and privacy guarantees of FIDO2 pose a real challenge to implementing secure and usable recovery solutions (also indicated by our respondents, see Section 3.4). The official FIDO Alliance recommendations [72] state that recovery can be done in a self-service manner using strong fallback authentication or an identity vetting procedure. Kunke et al. [123] evaluated 12 account recovery methods found in academic publications and in the wild (following the Bonneau et al. [38] framework) and concluded that FIDO backup tokens are the best of available methods. Notably, selecting a weaker recovery method opens the gate for FIDO2 downgrade attacks (i.e., forcing the

system to fall back to authentication methods with known vulnerabilities) described by Ulqinaku et al. [184]. Similarly to the migration use case, FIDO2 multi-device credentials [76] can be used for account recovery. An interesting solution was proposed by Visa Research [28], where a managerless signature group can be registered instead of a single authenticator, thus improving the usability of the multi-authenticator approach to account recovery.

3.3.5 Deprovisioning

The deprovisioning process marks the end of the identity lifecycle. Usually, the user's access is revoked and the account is deleted or suspended. Simultaneously, the account's active sessions are terminated and all bound authentication factors are deregistered. In the context of FIDO2, deregistration is as simple as removing the public key from the server storage. For the server-side credentials, storage clean-up is enough, however, in the case of discoverable credentials, they remain in the authenticator memory. Even though invalid, they take up space which, is an issue for devices with limited storage (e.g., for some Yubico hardware tokens the limit is 25 keys)[3]. Before the release of CTAP 2.1 [71] the only option to remove a key was to reset the authenticator, which removed all created keys. With the 2.1 version, an *authenticatorCredentialManagement* API was added to enable the removal of a single key. This API is an authenticator side operation that has to be triggered by the user via a CTAP client (e.g., specialized software provided by the device vendor).

In contrast to other procedures such as enrollment or authentication, deprovisioning may be considered rudimentary. Nonetheless, it was still included in the list of challenging processes according to our user study.

3.3.6 Usability

Regarding workforce usability, the integration decisions impact how FIDO2 is perceived. For example, the majority of modern devices already have built-in PAs, and

[3]https://support.yubico.com/hc/en-us/articles/360013647720-Security-Key-by-Yubico

thus employees' private devices can be leveraged (e.g., when the "bring your own device" policy is allowed). This approach was researched in the academic institution context by Weidman et al. [193] who found that employees perceived using private devices as unprofessional. On the other hand, workforce FIDO2 deployment in a small company was studied by Farke et al. [66]. Their findings indicate that most of the employees found key-based login as usable, however, several of them were unconvinced of security benefits and found password managers integrated with web browsers faster to use. Similarly, Farke et al. [65] examined passwordless authentication using Windows Hello for Business usability in the company setup and found it faster, more responsive, and convenient to use.

As mentioned previously, end-user usability is not the sole factor to be taken into consideration. Ease of deployment and post-deployment maintenance plays a significant role in the solution being successful. In particular, security-related products come with a demanding configuration process (e.g., Krombholz et al. [119] revealed TLS configuration issues, some of which are challenging even for security experts). This applies to FIDO2 as well, and like any technology, it struggles with the challenges of the early adoption stage (e.g., lack of technical know-how and examples). As noted by Alam et al. [23], the ongoing development of WebAuthn features and tools is critical. In recent years, the range of available tools and libraries noticeably increased (e.g., WebAuthn public resources [21]). Academics have also contributed to the development aspects of FIDO2. For example, Grammatopoulos et al. [187] provided an analysis tool for FIDO2 traffic. Regarding deployment, FIDO2 authentication is currently available in all major cloud providers, however, as noted by Gordin et al. [90], not all solutions provide an easy FIDO2 integration (e.g., Open Stack, one of the cloud orchestration platforms lacks support).

Despite the commendable efforts of the FIDO2 community to provide education regarding the protocol, issues related to usability and technical expertise continue to pose significant challenges for FIDO2 implementers (refer to Section 3.4.6).

3.4 Usability Study of FIDO2 Integration

To address our research inquiry, specifically, the challenges that discourage the integration of FIDO2, we conducted a user study and gathered input from professionals possessing practical expertise in IAM.

3.4.1 Study Design and Methodology

We prepared an online questionnaire following the human-computer interaction research guidelines described by Lazar et al. [126]. The online format provides us the flexibility to target our interest group remotely and globally. The questionnaire was written in English and implemented as a publicly available and anonymous Google Form. In collaboration with an industry expert, we formulated the study questions, subjecting them to an internal evaluation process. This evaluation was conducted by three autonomous senior academic researchers affiliated with Macquarie University. Additionally, we conducted a pilot test involving our internal research group, consisting of 15 subjects.

The questionnaire consists of 26 open and multiple-choice questions grouped into four sections. To allow unaided answers, we provided an "Other" option where appropriate. The full questionnaire can be found in Table 3.1. Depending on the question's nature, we used the following statistical tests to determine relations between variables in our study: Cramér's V (χ^2), Kruskal-Wallis (KW), and Pearson correlation. Unless otherwise indicated, Cramér's V was used. Presented results have a small Cramér's V effect unless otherwise noted. Results with $p \leqslant 0.05$[4] were considered statistically significant. We describe our findings in the sections below and provide an exhaustive list of statistical tests' results in Appendices A.3 and A.4.

[4]A p-value measures the probability of obtaining the observed results, assuming that the null hypothesis is true.

Questions
1. What is your role within your organisation?
2. What is your experience in the IT field?
3. What is your experience in the Identity and Access Management (IAM) field?
4. Are you currently involved in an IAM project within your organisation?
5. What is the industry of your organisation?
6. How big is your organisation in terms of number of users (employees, contractors, etc) that sign in?
7. Is your organisation (along with its customers and users) present in multiple countries? How many countries?
8. Does your organisation operate in a regulated environment (...) that requires the use of MFA or passwordless?
9. Does your organisation use on-premise and cloud applications?
10. In your opinion, is your organisation ready to move from passwords to passwordless authentication?
11. Is your organisation already using an existing passwordless authentication solution?
12. In your opinion, is it important to adopt a fully passwordless authentication?
13. Have you been involved in evaluation (...) of any solution for passwordless authentication?
14. How well do you know FIDO2 technology for passwordless authentication?
15. Have you been involved in evaluation (...) of any solution based on the FIDO2 technology?
16. What is your preferred way of implementing FIDO2 passwordless authentication within your organisation?
17. What is your preferred model to introduce FIDO2 passwordless solution: software or ... SaaS?
18. Are you currently adopting a solution based on the FIDO2 technology ... (in production)?
19. Which of the following FIDO2 passwordless authentication features are you familiar with?
20. What are the key ... challenges which you observed during ... adoption of a FIDO2 passwordless authentication solution?
21. Which part of the user ...related process is difficult to change ... to work with FIDO2 passwordless authentication ...?
22. Which scenarios do you consider most challenging for adoption of a FIDO2 passwordless authentication?
23. Which authenticator type is considered strategic for your organisation?
24. In your opinion, which authentication channels can be problematic for FIDO2 passwordless authentication?
25. Please briefly describe the use cases that are most important ... to adopt FIDO2 passwordless authentication?
26. ... what other obstacles (...) have you observed that would prevent ... deployment of FIDO2 passwordless authentication ...?

Table 3.1: Usability study questions.

3.4.2 Recruitment and Participants

We opened the user study in the first quarter of 2022 and performed an advertising campaign throughout the year. The study target group is individuals working towards implementing cybersecurity solutions. Importantly, to increase the completeness of the study, we invited professionals from different roles including decision-makers, managers, SMEs, and developers. The call for participation was promoted through leaflets, personal appearances at cybersecurity conferences and meetups, personal contacts, and public posts on social media such as LinkedIn, related discord channels, and the FIDO Dev (`fido-dev`) email list.

We recognize that our recruitment strategy may introduce a bias towards individuals and organizations with a specific interest in FIDO2. Nevertheless, given that the primary objective of this study is to gain insights into FIDO2 perceptions among practitioners, rather than assessing FIDO2's widespread perception, we claim that this bias does not adversely impact our findings.

3.4.3 Ethical Considerations

The study was approved by the Science & Engineering Subcommittee at Macquarie University (reference number 520231270444606). We endeavored to ensure that collected data does not contain any private or sensitive information. Participation in the study was voluntary and we did not require or record any form of identification. All participants were informed about the study terms before engaging with the questionnaire.

3.4.4 Participants Profile

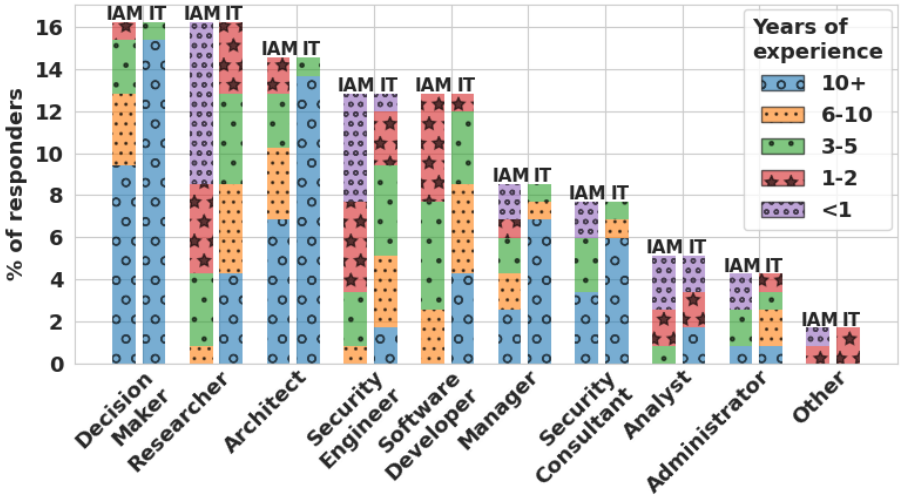


Figure 3.2: Years of experience in Identity and Access Management (IAM) and Information Technology (IT) by profession.

Firstly, we analyzed the professional profile of our respondents. As shown in Figure 3.2, our participants hold a diversity of roles in the enterprise hierarchy (no role had more than 16.2% of responses). Over 50% of the respondents identified themselves as senior IT personnel (i.e., 10+ IT experience) and being involved in IAM for at least 3 years. Moreover, over 40% of the respondents actively participate in an IAM project at their organization. The roles diversity, strong IT and IAM background and hands-on experience of the participants support the credibility of our data. In regards to professions, the frontliners in investigating and adapting new technology

(i.e., decision makers, researchers, and architects) were the most numerous groups. Moreover, we observed that a significant proportion of respondents are technical and strategic leaders (decision makers, architects, and managers). They stand for over 40% of answers and qualify as highly experienced specialists with over 90% having more than 10 years experience in IT and 50% more than 10 years in IAM.

The study participants represent a diverse spectrum of organizations including large global enterprises. Over 50% of respondents work in organizations that provide IAM services for more than 1,000 users and over 35% of them are multi-region companies (i.e., operating in 2 or more countries). In total, over 60% of the data set contains multi-country organizations. In terms of industries, over 75% of responses were submitted by employees of three sectors: Technology, Research and Education, and Finance and over 50% answers indicate companies operating in regulated environments (e.g., PCI DSS or HIPAA). Interestingly, the majority of organizations (almost 80%) use both cloud and on-premise applications. The above observations derived from our study data (presented in Figure 3.3) strongly suggest that a significant proportion of our participants operate in highly demanding environments which impose additional requirements for authentication.

3.4.5 Passwordless Authentication

According to our respondents, only 45% of their organizations are already using or are ready for migration to passwordless authentication. The proportion was highest for the Technology sector (61%), followed by Finance (44%) and Research and Education (38%). Surprisingly, over 30% declare that their organization is not ready to move away from passwords. In terms of deployments, almost 54% of respondents admit that their organization does not use any passwordless authentication solution. Among existing production systems the preference splits almost equally between vendor-provided software(18.5%) and vendor-managed cloud solutions (19.3%). In-house built solutions were present in only 8.4% answers. The details for each industry can be found in Figure 3.3. Almost 37% of respondents admitted that they were involved in passwordless

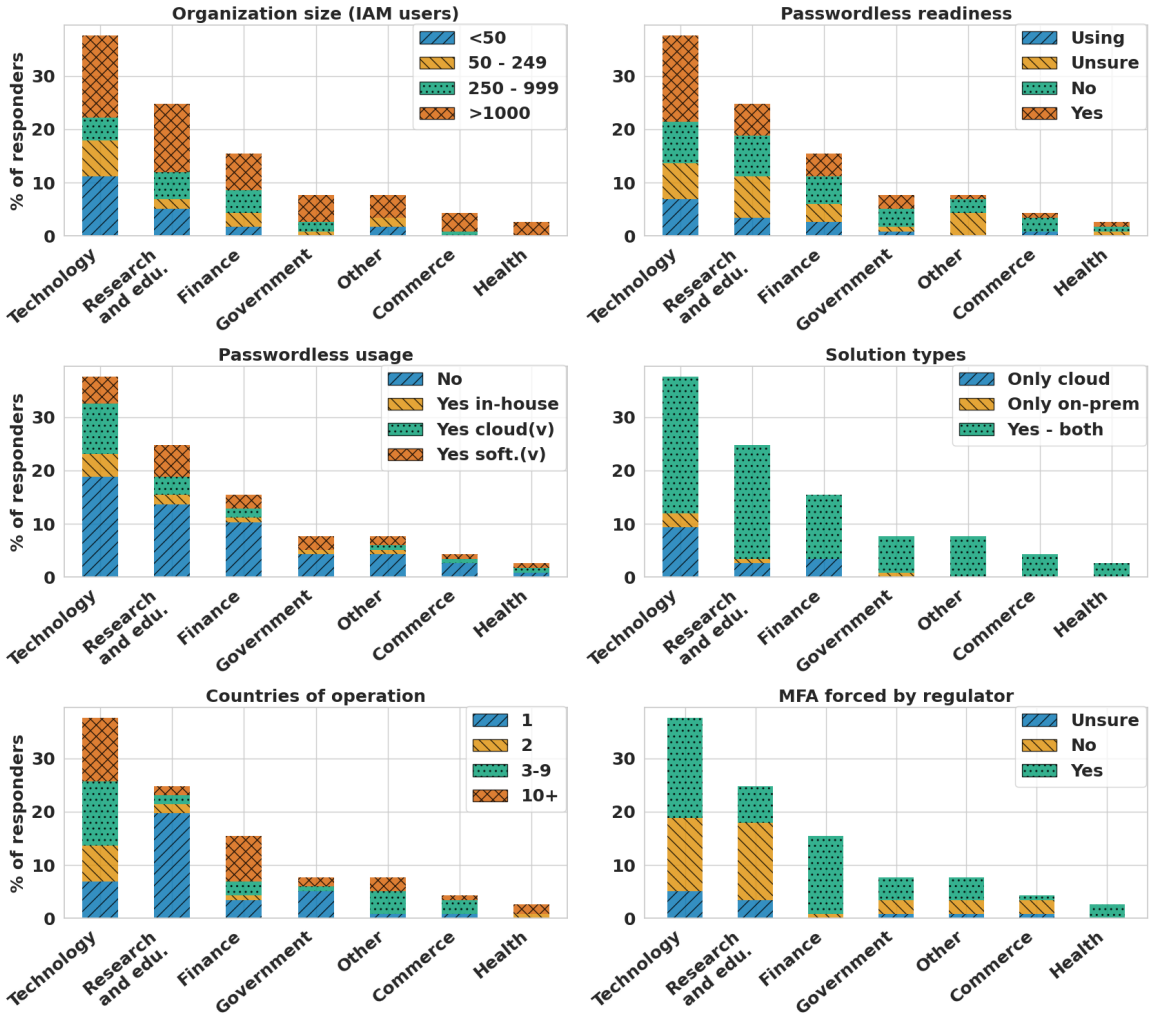


Figure 3.3: Organization and passworldess authentication related responses by industry.

solution evaluation. One-third participated in the evaluation of a FIDO2 solution. Notably, 12% of respondents who did not evaluate passwordless solutions, answered that they evaluated FIDO2. We hypothesize that FIDO2 may have been evaluated as a 2FA (second factor) in solutions that still rely on passwords as the first factor. The data presented above (as depicted in Figure 3.4) indicates that FIDO2 has progressed into the "Early Majority" phase of the innovation adoption lifecycle.

The last question in this section measured the participants' perception of the importance of passwordless authentication on a 10-point Likert scale (see Figure 3.5). Generally, passwordless authentication is perceived as important with the majority of

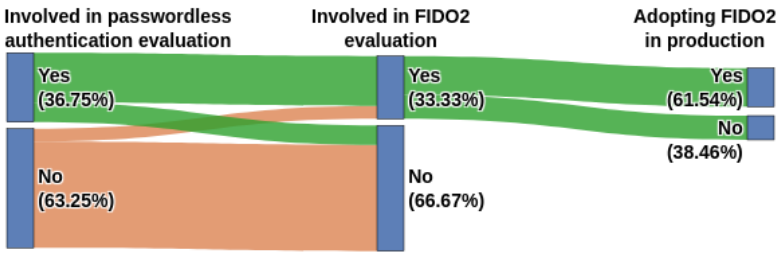


Figure 3.4: Proportions of respondents evaluating passwordless authentication and FIDO2, and adopting FIDO2 in production.

answers above the neutral threshold (i.e., 5 on the scale) and 43.7% of answers stating it is extremely important. In terms of professions, we noticed a significant increase in answers towards high importance from decision makers, security consultants, and architects. We hypothesize that these roles need to follow new technologies and threats to direct work in their organizations, and thus are more convinced and aware of the importance of passwordless authentication. Interestingly, managers' and researchers' answers were less extreme. We speculate that their role involves consideration of a wider context beyond IAM, which might decrease the apparent importance of passwordless authentication. Surprisingly, the analysis by industry (right two panels of Figure 3.5), showed that government employees are highly convinced of the importance of passwordless authentication.

Regarding statistical analysis, we identified the following relations. Managers were found to not confirm the passwordless readiness of their organizations, whereas decision makers more frequently answered positively. Similarly, the technology industry stands out in indications of passwordless readiness. Organizations operating in non-regulated environments were linked with not being ready for passwordless. Passwordless importance was found to be related to readiness (KW small effect). Experienced employees (IT 10+) and in particular decision makers, architects, and security consultants were found to more frequently use passwordless solutions. Similarly, the experienced personnel more frequently participated in the evaluation of passwordless solutions. Surprisingly, participants with less than one year of experience as well as those working for the government tended to have been involved in passwordless evaluation.

3.4.6 FIDO2

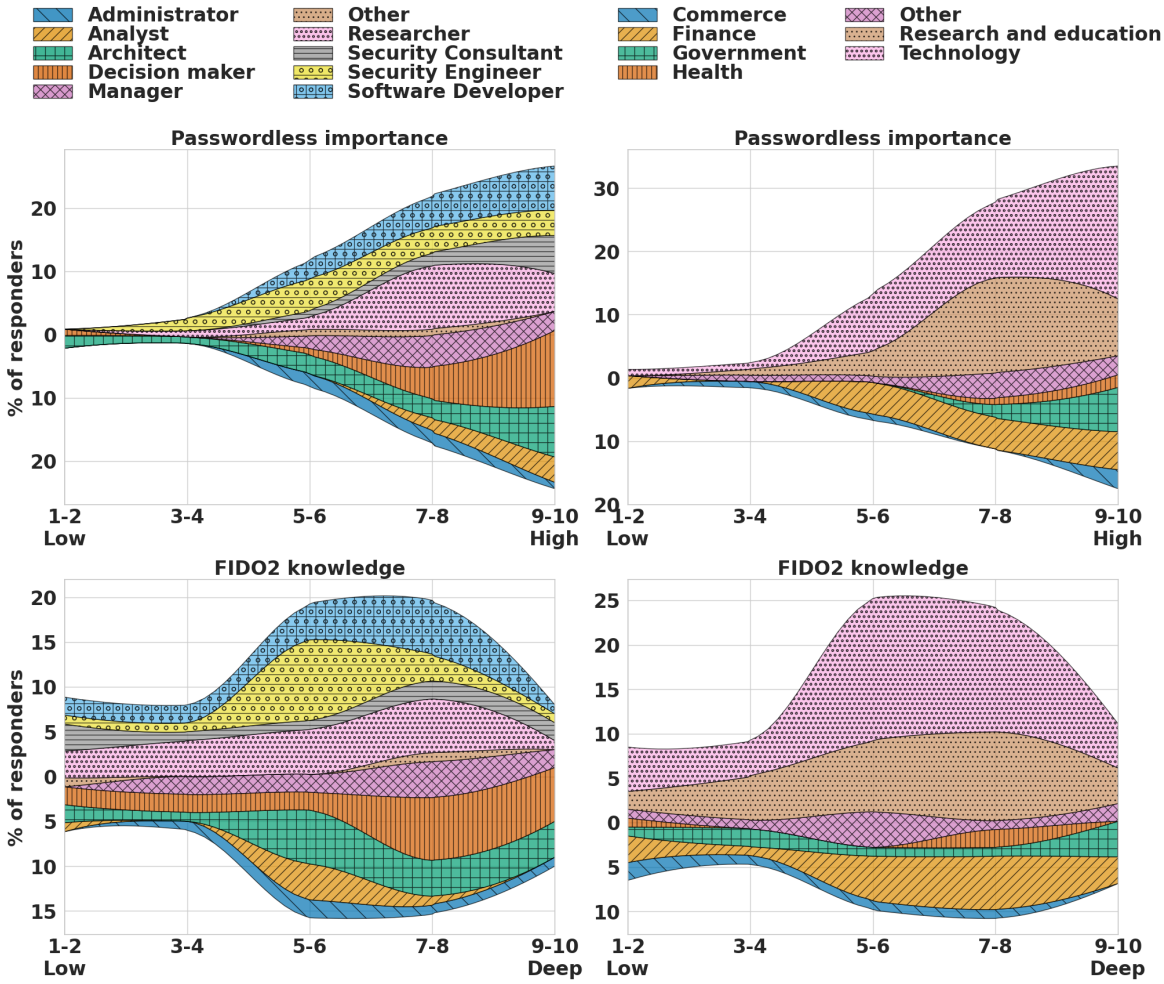


Figure 3.5: Passwordless authentication importance and FIDO2 knowledge questions by profession.

Knowledge

In Figure 3.5 (second row), we depict the respondents' self-assessed levels of FIDO2 knowledge. We present the collected data while considering both the respondents' professions (displayed in the first column) and their respective industries (displayed in the second column). The FIDO2 knowledge distribution peaks exactly in the middle with a median of 6. We analyzed the knowledge distribution per profession and industry and found that the medians of management roles (i.e., decision maker and manager) are 1 point above the distribution median. Notably, we observed lower medians (5)

for the engineering roles (i.e., software developer, security consultant, and analyst). In terms of industries, the employees of typically regulated industries recorded higher median values: Government (7), Health (7), and Finance (6.5). The lowest median of 4 was found for the Commerce industry. Passwordless importance and FIDO knowledge show a positive correlation ($\rho = 0.2, p = 0.03$). The FIDO knowledge variable was found to relate to IAM experience with a large effect (KW), and to having evaluated passwordless (medium effect) and FIDO (large effect) (KW). A medium effect relation was also found with the number of countries where an organization operates (KW), and a large effect relation with ongoing FIDO deployments in production (KW).

Evaluation and Production Deployments

Respondents who participated in the evaluation of any FIDO2 solution were asked additional, more detailed questions. A total of 39 participants (33%) qualified for this section. Almost half of the respondents are decision makers and architects, and almost 80% are highly experienced in IT (10+ years) and over 40% in IAM (complete profile in Appendix A.1). Notably, 62% of this group is in the process of adopting FIDO2 in production (see Figure 3.4). The data analysis shows medium effect relations between participants experienced in IAM and FIDO2 evaluation, however, the opposite relation (i.e., not participating in FIDO2 evaluation) was found for employees in IT with 10 or more years of experience. Notably, a relation was found between the positive answers regarding the evaluation of FIDO2 and government sector employees. We found relations with ongoing FIDO2 deployment, many of which have a medium or large effect. Firstly, the relation with top experienced IAM personnel had a large effect, whereas participants with 1-2 and 3-5 years of experience had a medium effect relation to *not* having FIDO2 deployment in production. Interestingly, decision makers were the only profession, for which we found a relation with FIDO2 deployment in production (medium effect), and only one use case, "privileged accounts", showed a relation. For negative answers regarding the production deployment of FIDO2 we found a medium effect relation with single-country organizations and the research and education industry.

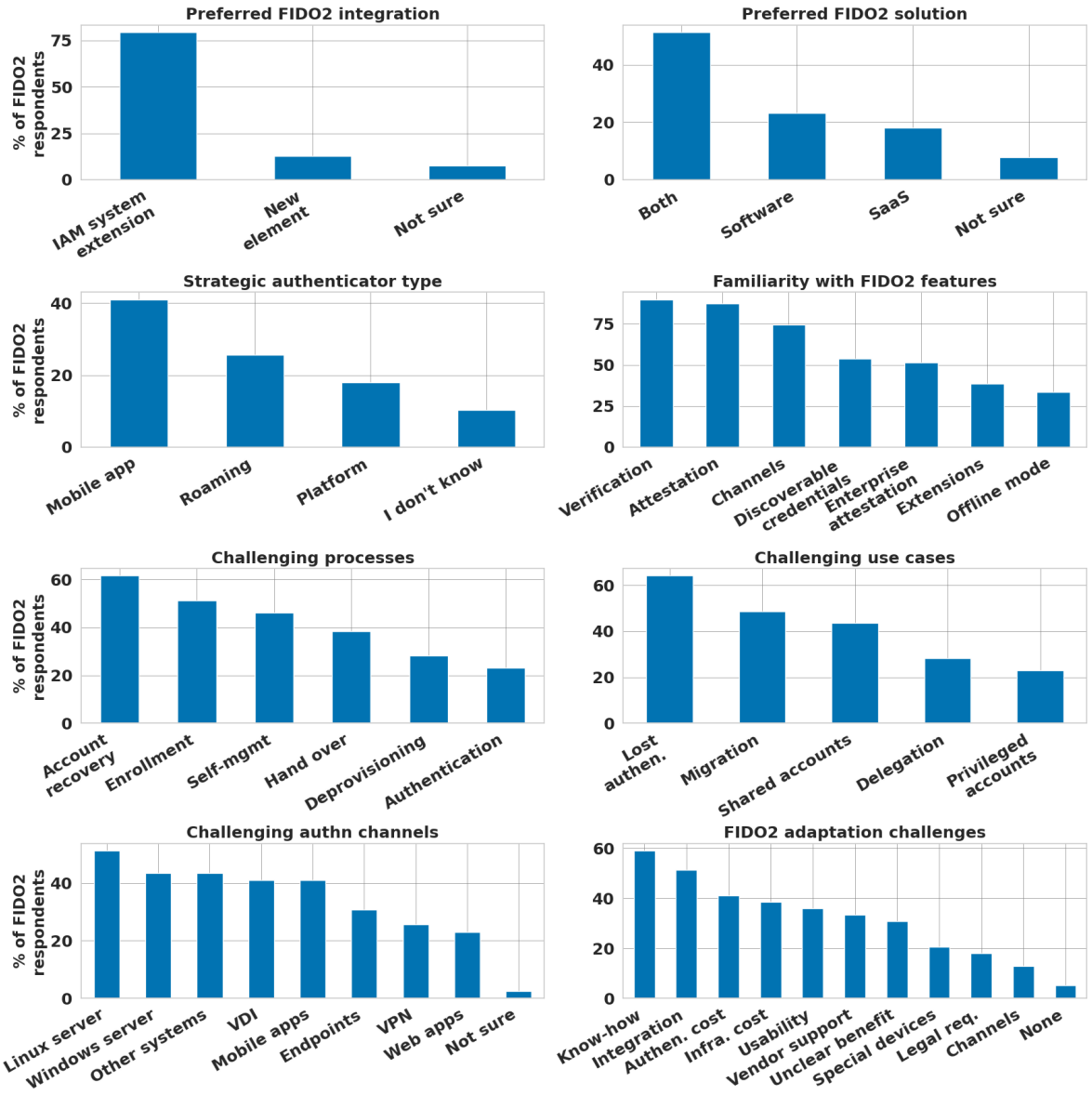


Figure 3.6: FIDO2 preferences and challenges (only respondents involved in FIDO2).

Preferences and Familiarity

Figure 3.6 summarises responses to questions on FIDO2 preferences, familiarity, and challenges. The respondents show a clear preference for FIDO2 integration as an existing IAM solution extension (79%). In terms of the deployment model, 51% of participants prefer to have both software and vendor-managed solutions (e.g., in cloud), with 23% preferring software only and 18% purely SaaS (Software as a Service). The SaaS option had a medium effect relation with the technology sector, participants with less than 1 year of experience, organizations with on-prem and cloud, and those

with only cloud presence. Regarding the preferred authenticator, we observed mobile applications as the most desired type of authenticator (41%). The second pick was roaming authenticators (26%) followed by platform authenticators (18%). Notably, hardware roaming authenticators had a medium effect relation with staff with 1-2 years of IT experience and with organizations below 50 users. On the other hand, platform authenticators were related (medium effect) to organizations not yet ready to migrate and those with on-prem presence only. Interestingly, we found a few medium-effect relations to "I don't know which authenticator type" and "Neither authenticator type" on authenticator preferences. For example, the technology industry, the architect profession, and the deployment of both on-prem and cloud. This suggests insufficient knowledge about authenticator types.

In addition to preferences, we measured our participants' familiarity with FIDO2 features. Clearly, user verification and attestation are well-known features with 90% and 87% of responses respectively. A slightly less recognized feature of FIDO (74%) is the support for various transport channels. Surprisingly, two features, which are particularly useful in the enterprise context: discoverable credentials (a.k.a. resident keys) and enterprise attestation, are only known to about half of the respondents (54% and 51%). The least known features of FIDO2, according to our respondents, are FIDO extensions (38%) and Offline authentication mode (33%).

Challenges in Adaptation and Processes

To understand FIDO2 challenges, we asked our participants which aspects of passwordless FIDO2 authentication are considered difficult. We organized questions into four categories: adaptation, processes, use cases, and authentication channel challenges. For the questions in this group, participants could choose to submit open responses instead of the suggested responses, which we discuss in Section 3.4.7.

The main reason why FIDO2 adaptation is challenging according to our respondents is a lack of knowledge and know-how (59%). Over half of the participants (51%) identified integration with their systems as being complicated. For the third and fourth place, the respondents selected the cost of authenticators and infrastructure (41% and 38% respectively). FIDO2 usability is considered a challenging factor for over one-third

(36%) of participants and 33% believe that vendor's support is insufficient. We found that the unclear benefits of FIDO2 authentication are a noticeable adaptation challenge (31%). The requirement for a specialized FIDO2 device (i.e., an authenticator) is a challenging factor for only 20%. Legal requirements and connectivity channels were identified as the least challenging (18% and 13% respectively). Our analysis shows that the "no challenges" answer relates to less experienced participants (below 1 year of IAM experience with a large effect and in 1-2 years in IT). Similarly, a relationship was found between the security consultant role and SaaS as the preferred FIDO2 deployment model. Notably, connectivity channels as a challenge were found related to single-country organizations and participants who are unsure about the preferred deployment model. Furthermore, we found a relationship between the challenge of legal requirements with participants with 1-2 years of experience in IT.

For the identity lifecycle processes, FIDO2 was found the most challenging for account recovery (62%). Over half (51%) of respondents marked the enrollment flow, and 46% found self-management with FIDO2 challenging. The initial and final processes (i.e., hand-over and deprovisioning) were only selected by 38% and 28% participants respectively. Surprisingly, the main functionality of FIDO2 (i.e., authentication) was identified as problematic to integrate into the processes for only 23% respondents. The only significant relationship found for challenges with identity lifecycle was between authentication processes and organizations below 50 users.

Challenging Use Cases and Authentication Channels

Regarding challenging use cases, the majority of respondents (64%) selected "lost authenticator", followed by "authenticator migration" (49% of answers). Similarly, shared accounts were identified as a challenge by 44%. Less than 30% of respondents believe that authentication delegation (28%) and privileged account (23%) use cases are challenging. Our analysis found that the "none" answer was related to software developers, organizations operating in multiple countries (3-9) (large effect) and participants with 3-5 years of IAM experience. The delegation use case showed a relation with the research and education industry. Interestingly, the migration use case had a relation to uncertainty about passwordless readiness, and challenges around privileged

accounts were found to be related to experienced IAM employees and participants working on the production deployments of FIDO2.

The last questions' category focuses on the challenging authentication channels. We found that authentication to remote servers is perceived as problematic. Linux servers were identified as the most challenging environment (51%), followed by Windows (44%) and other servers (44%), with only a few percentage points less (41%) for both VDI and mobile applications. Access to endpoints was identified as challenging by 31% of participants and VPN authentication by 25%. Notably, the main target of FIDO2 authentication (i.e., web applications) was selected by 23% of the respondents. We found medium effect relations between the "not sure" answer and both managers and participants with 6-10 years of IAM experience. Additionally, a relation was found with software as a preferred deployment model. The VPN channel was related to participants with 3-5 years of IT experience and participants unsure about their company's readiness for a passwordless migration. Furthermore, the mobile channel was related to organizations that already are using a passwordless solution.

3.4.7 Free Text FIDO2 Questions

Considering the wide scope of possible challenges with FIDO2 adoption, we provided an option to submit free text answers which we discuss below. Regarding the challenges to FIDO2 adoption, respondents most frequently addressed the incompleteness of the FIDO2 environment. Firstly, technical issues were drawn to our attention. The following opinions: *"missing/incomplete FIDO2 mobile solutions"*, *"sync between multiple platforms (ms/apple/google)"*, *"lack of authenticator support and availability of authenticators with biometrics"* suggest that the existing implementations do not meet industry expectations, and thus make it challenging to roll out a FIDO2-based solution. Similarly, one of the respondents addressed technical challenges with the edge cases (*"support legacy applications and use cases where USB/NFC/BLE cannot be used"*). We also received an observation about FIDO2 documentation (*"limited documentation for FIDO2 server implementors"*) and challenges in the correct delivery method (*"secure delivery of a roaming authenticator"*).

Respondents spotted challenges in aligning organizations and users towards FIDO2. In particular, two answers (*"competing priorities"* and *"unclear benefit of FIDO2 vs Push Notifications"*) suggest that the idea and importance of secure phishing-resistant passwordless authentication are not yet commonly known. Notably, complex enterprise environments can impact how challenging the FIDO2 integration is. For example, one of the respondents answered that their organization has a *"requirement for both mobile devices and dedicated authenticators"*. Regarding end users, user experience was indicated as a challenging factor. One respondent identified user interface messaging as an issue (*"it is confusing for end users at first, all these different prompts because different browsers present differently."*). The second opinion described a specific flow that lacks smooth user experience (*"when integrating with OpenID Connect, the user experience involves a switch from mobile app to browser for authentication"*).

The responses collected for organizations' most important FIDO2 use cases revealed that completeness and unification of the FIDO2 environment play a critical role. In particular, the support for various authentication channels was repeatedly mentioned (*"VPN, access to a web app, mobile app, access to servers, critical infrastructure like DNS, G Suite, AWS"*, *"access to desktop computers and laptop devices; Access to VDI"*). Additionally, respondents outlined a unified user experience as a significant factor in their organizations (*"same/similar UX ... as password managers, which users are familiar with"*, *"FIDO2 needs to be ubiquitous across platforms..."*). Another use case identified by our participants is device support and BYOD (bring your own device) policy. For example, *"ease of use through manufacturer level support on all employee devices, ..."* as well as *"FIDO2 BYOD solutions for mobile"* are considered to be crucial use cases. Surprisingly, only one opinion mentioned security features as important use cases (*"phishing resistant MFA"*).

In the final open question, we captured participants' opinions about other (i.e., not mentioned before) obstacles and challenges that could prevent or negatively impact FIDO2 deployment. The majority of our respondents' answers point to human-related issues. The first one addresses user adaptation challenges. For example, *"the resistance of the users/administrators to new technology"*, *"user awareness"* and *"relatively new,*

... it's hard for most people to get over the hump benefit of PKI with binding on the authenticator" suggest that understanding of FIDO2 technology is not sufficient, and thus difficult to successfully incorporate in the organization's security suite. Moreover, the following responses *"lack of a champion in a leadership role"* and *"complicated for non-technical users to understand"* reveal that organizations struggle with passwordless migration process due to missing know-how. Two of the respondents expressed security concerns related to the quality of authenticators (*"bugs due to local software errors on devices, producing inconsistencies in authentication reliability"*) and the latest advancement in the FIDO2 environment (*"Apple's implementation of Passkeys is a concern without DPK or attestation..."*). One of the respondents argued that some organizations already use passwordless solutions and do not see incentives in migrating to FIDO2 (*"US Federal Government already has the PIV/CAC. They don't see much value in FIDO2, yet"*).

3.5 Main Takeaways

Statistical analysis as well as free text answers provide a clear picture of the challenges with FIDO2 integration. One of the major factors that discourage FIDO2 integration is know-how and toolset. Our respondents believe that even though their general knowledge of FIDO2 is satisfactory, practical knowledge and integration paths are still missing and pose a major obstacle. In particular, adequate handling of processes such as account recovery or enrollment, and edge cases like lost authenticator, were found challenging. Fundamental processes such as registration and authentication were also found to pose significant challeges by the study. Solution costs and missing native integrations were spotted as major integration blockers. In particular, integration with commonly used servers (e.g., Windows or Linux) poses a significant challenge for enterprise deployment. Furthermore, the respondents clearly articulated that usability, more precisely, differences in the presentation (UI) and process (UX) pose a challenge for the integration into users' daily routines.

In addition to challenges, our study identified the most preferred FIDO2 adaptation

model (i.e., an extension of an existing IAM platform, which can run as both SaaS and a software solution) as well as the preferred FIDO2 authenticator (i.e., mobile application). We learned that even though respondents are familiar with the fundamental FIDO2 features, the more advanced properties are less known, which could explain why the unclear benefits were perceived as a challenge.

The insights mentioned above provided a baseline for the development of our privacy-preserving system, as elaborated in Chapter 5. Notably, the identified preferences were integrated to enhance the design of our system, positioning it as a natural extension of the existing IAM protocols primarily tailored for mobile devices. Moreover, the potential hurdles associated with adaptation were effectively tackled through two key strategies: first, the minimization of requisite expertise (manifested in the delegation of all privacy-preserving mechanisms to our implementation); and second, the establishment of a seamless integration process (characterized by pluggable front-end integration and streamlined back-end integration).

3.6 Conclusion

In this work, we asked IT specialists and FIDO2 implementers about their perspectives on the challenging aspects of FIDO2 adaptation. Our initial review of use cases shows that the FIDO2 ecosystem, even though having an admirable coverage of the use cases, still lacks solid and production-ready solutions for certain processes (e.g., account recovery). Our findings were confirmed by the user study, in which we managed to identify and order the most challenging aspects of FIDO2 adaptation (i.e., lack of know-how, cost, and usability). We believe that our results, backed by the specialists' feedback, provide clear directions for future FIDO2 developments. In particular, we anticipate that our study will be utilized as a guiding reference for usability researchers and the FIDO community, as they work towards advancing the FIDO2 environment, much like the usability enhancements we achieved within our system. In the next chapter, we examine privacy properties of FIDO2 and report a surprising finding that undermines FIDO2 privacy guarantees.

4

FIDO2 Privacy - Timing Attacks

This chapter is adapted from the work titled "How Not to Handle Keys: Timing Attacks on FIDO Authenticator Privacy" published in the Proceedings on Privacy Enhancing Technologies 2022 pp. 705–726. Authors: M. Kepkowski, L. Hanzlik, I. Wood, M. A. Kaafar*

In the preceding chapter, we conducted a comprehensive evaluation of the usability of FIDO2 integration, identifying key challenges that we scrupulously addressed within our design of a privacy-preserving system for IAM. In this chapter, our investigation delves deeper into the FIDO2 protocol and its suitability for accommodating privacy-enhancing extensions. FIDO2 is inherently designed to incorporate privacy-preserving mechanisms, safeguarding against the correlation of user accounts - precisely aligned with the requisites of our system. However, throughout the course of our research, we discovered a novel attack that undermines these assumptions. Left unaddressed, this

vulnerability has the potential to compromise our efforts to improve the privacy of IAM systems. Therefore, in this chapter, we investigate the attack and propose mitigations, some of which were implemented in popular FIDO2 clients (e.g., web browsers).

FIDO2 authenticator, like any computational unit, possesses a set of operational properties (e.g., power consumption curve) that impact the environment. For example, when dealing with complex mathematical problems involving numerous additions and multiplications, the chip's operation emits operating signals. Cryptographic operations, such as signature generation and data encryption, are particularly computationally intensive and leave a distinct footprint. Properties such as power consumption, emitted electromagnetic field, or processing time have the potential to leak information about the chip's operations. Although measuring these properties often requires significant resources and expertise, doing so can lead to a complete compromise of the security and privacy scheme.

These attacks, collectively known as side-channel attacks, aim to extract secrets through physical signal measurements. For example, Lomne et al. demonstrated in their study[14] that precise measurements of electromagnetic radiation during ECDSA signatures enabled the recovery of private keys from one model of the hardware authenticator. It is crucial to emphasize that side-channel attacks not only compromise security but also have the potential to undermine privacy-related attributes, which holds true for the attack presented in this chapter.

In this context, we introduce a novel side-channel attack on the privacy aspects of the FIDO2 protocol and proceed to assess its implications for both commercial authenticators and FIDO2 clients. We propose mitigative measures and collaborate with vendors to implement remedies for their products. Beyond our contribution to improving the overall privacy of the FIDO2 ecosystem, we have also applied our insights to formally analyze the privacy attributes of our proposed system, as detailed in Chapter 5.

4.1 Introduction

Password-based authentication methods are infamous for their security weaknesses [181; 113], which led to the adoption of second factor authentication such as software based approaches like Google Authenticator[1] and Duo Security[2], and hardware based tokens like Yubico Yubikey[3] and HyperFIDO[4]. Authentication tokens provide a challenge-response based protocol using a standard specified by the FIDO Alliance [39] called FIDO2 (Fast Identity Online), as a successor of UAF (Universal Authentication Framework) [127] and U2F (Universal 2nd Factor) [174].

The authenticator/token holds a secret key that is used to authenticate against a public key bound to the user's account during registration. FIDO2 with its Client to Authenticator Protocol (CTAP) [4], combines both and is the state-of-the-art standard for user-side authentication which is complemented by the W3C WebAuthn specification [121]. A comprehensive description of the FIDO2 protocol can be found in Chapter 2.

The adoption of FIDO2 is driven by the support of major service providers on both mobile and desktop platforms (Android, IOS and Windows [3][5][8]) and industry implementations (some examples are: US login.gov page [6], ID intelligence suite from VISA [7], NHS enhanced login [12]). The main goal of the FIDO2 protocol is to mitigate known problems with existing authentication mechanisms. In particular, the challenge-response design of the protocol protects users against replay and credential theft attacks. The protocol is known to be immune to database leaks since the authentication servers store only the public keys of users [15]. The protocol is also designed to protect the privacy of users by preventing the linkage of accounts for which the same authenticator was registered (see "Privacy considerations for authenticators" section of WebAuthn [15]). While most of the research efforts into FIDO and related technologies have focused on providing security guarantees, in particular with strong and

[1] https://g.co/2step
[2] https://duo.com
[3] https://www.yubico.com/
[4] https://www.hypersecu.com/

robust end-user authentication, considerations of privacy have received less attention from the security community.

The linking of a users' accounts across internet platforms, our focus here, poses both common risks such as undesirably targeted recommendations and advertising [43] and other purposes [84; 136; 162], as well as more dangerous risks such as enabling actors with malicious intent towards an individual such as criminals or unfriendly state actors. FIDO2 authenticators are one of the proposed solutions to these problems and are being actively promoted by security oriented companies. For example, Microsoft, Yubico and Google run programs to empower at-risk individuals by ensuring strong and privacy preserving authentication via FIDO2, and in 2021, they distributed 35k tokens [16; 10].

In a regular login/password based authentication scenario, a user can aim at protecting their privacy across services by creating a service-specific identity, e.g., by using different e-mail accounts during registration and different login/password information. Even though the method is not perfect, it allows some protection against providers trying to easily link user accounts across different services. Note that a malicious service can always use other metadata (e.g., tracking cookies or geolocation) to link user accounts and protection against such attacks is an orthogonal problem. Unfortunately, the introduction of any second factor for authentication increases the privacy threat of linking identities across services. In particular, binding the same hardware token with unique public keys to the user's accounts connects the identity of the user across the various services. The FIDO2 protocol seeks to mitigate this risk through privacy preserving registration and authentication algorithms.

Unlinkability across services together with user authentication is one of the two *fundamental* properties of the FIDO2 authentication mechanism [70]. In fact, the FIDO standard recommends that hardware tokens generate unique public keys for each service. Unfortunately, in practice this solution does not scale well if the device has only a limited amount of secure memory to store secret keys. This issue is addressed by the consideration of *non-resident* keys, i.e.: keys for which the secret key is recomputed during the authentication process and not stored on-device. A common approach to

implement this is to use the *key-wrapping* technique [62]. The idea is for the server to provide the token with a ciphertext containing the secret key that can be used for authentication. This secret key is then decrypted using a master key stored on the device. We show that wrong processing of key handles can lead to a significant difference in execution time depending on what data is used. In particular, for vulnerable authenticators there is a time difference between processing key handles from a given authenticator and those that are not. If the attack is performed in the context of a valid user authentication, this provides the attacker with an approach to link a candidate key handle (and its associated user account) with the key handle and account to which the user is authenticating. Note that Relying Parties, the services to which a user authenticates, may stipulate that a *resident key* is required, which quickly consumes the memory of authenticator tokens and thus reduces their utility, however renders the site immune to this attack.

We focus on a remote form of the attack in which malicious software running on the users hardware is not required, but the attacker must have the capability to modify FIDO communications and to time FIDO calls to the authenticator. In practice, the timing must be done by malicious JavaScript code, so either ownership of JavaScript related to authentication or the ability to inject JavaScript is required. Surprisingly, numerous parties that interact with FIDO flow have these capabilities. We present a list of such actors with corresponding motivations and attack scenarios, including FIDO service providers, web proxies and adversaries exploiting XSS (Cross Site Scripting) vulnerabilities. Note that malicious code running on user hardware with a CTAP API (e.g., malicious apps or compromised browsers on Linux, Windows and MacOS, a rather stringent requirement) can execute silent CTAP authentications to both probe authenticators to link key handles and actually authenticate with user services when key handles are known.

The FIDO2 specification stipulates that an authenticator must perform a user presence check during the first phase of authentication, adding an indeterminate delay to the corresponding CTAP call. We propose two approaches to mitigate this and thus allow the timing attack to proceed. Our first proposal utilises multiple key handles in a

single CTAP call that requires only a single user presence check. In this way, the indeterminate nature of the user response can be averaged over multiple calls rendering the timing difference measurable. This attack is close to undetectable but requires clients (typically browsers) to allow long lists of key handles in a single CTAP call, which is the case for several major web browsers. Our second proposal uses audio recording to identify the point in time at which a physical button on the authenticator is pressed, thus circumventing restrictions in some clients on the number of handles in a single CTAP call. We demonstrate that the button click is easily detectable in a typical home environment.

We perform experiments with popular FIDO2 clients running on different operating systems and authenticators from various manufacturers. Our analysis reveals that two of the eight hardware authenticators we tested are vulnerable to our timing attacks. Moreover, our experiments show that the attacks can be executed remotely (for example as an external web service) through popular web browsers (Chrome, Firefox, Safari and others). We note that the FIDO security measures document stipulates in [SM-29]: "No leakage of secret information to remote entities via variation of operation execution time" [70]. The vulnerable tokens, which have both passed a security certification on L1 by the FIDO Alliance, clearly fail in this respect. Our examination of the certification procedures revealed that only L3 certification provides in-depth and on-device testing (i.e., empirical tests executed on the authenticator), however at the time of writing there are no authenticators with L3 certification.

One might hope that the proposed attacks can be easily mitigated by providing a firmware update for vulnerable authenticators. Unfortunately, to increase security most come without an update functionality. As a partial solution that prevents remote attacks, we propose and discuss ways for browser providers to mitigate the attacks. Note that the attack would still be possible from software running on the users device, hence replacing vulnerable tokens is the only complete solution.

To better understand the extent of the threat, we developed and ran a web crawler, which gathered information about FIDO2 implementations in the wild. We collected 684 records of FIDO2 in Javascript sources from high traffic websites. The results

showed that the only the passwordless implementation used by Microsoft required resident keys and could not be used for deploying the attack. The remaining implementations use FIDO2 as a second factor with non-resident keys and are thus in position to break the privacy of vulnerable FIDO authenticators.

Our approach leverages previously undiscovered weaknesses in the FIDO2 specifications and the ways that those specifications are implemented. Drawing on well-established techniques to exploit side-channel vulnerabilities such as improper error handling and execution time differences, we succeeded in finding a novel and easy to execute chain of actions that allow an adversary to learn additional information from vulnerable authenticators. We discuss the consequences of our findings as well as lessons applicable to any authentication system.

The contributions of this chapter can be summarized as follows:

1. We present a timing attack on the FIDO2 protocol that enables attackers to link user accounts, a serious privacy breach.
2. We demonstrate a remote execution method that allows the attack to be performed by website JavaScript code.
3. Two of the eight hardware token authenticators we tested were vulnerable out of a field of 111, indicating a substantial public privacy concern.
4. We proposed mitigation measures for FIDO clients that prevent the remote form of the attack and for FIDO authenticators. We notified relevant vendors and we participated in the mitigation design.
5. We surveyed 1 million high traffic web sites and found 684 FIDO authentication deployments, of which almost all allow *non-resident* keys and are thus exposed to our attack.

4.2 Adversarial Model and Attack

FIDO2 authentication scenarios require a certain level of flexibility (e.g., allowing users to register multiple authenticators under the same account). The FIDO2 protocol provides mechanisms for additional functionalities to support these scenarios. Two of

them, which are particularly useful for our attack, are silent authentication and the *allowCredential* list. Implementation flaws of these functionalities, described below, lead to a remote channel to measure authenticator execution time. This capability enables an adversary to detect differences in key handle processing times. Notably, the most popular mechanism to store key handles (non-resident keys), if incorrectly implemented, can allow a timing side channel to identify key handles that are associated with a given authenticator. Given that authenticators are typically used by single individuals, the combination of the elements mentioned above creates a remote attack vector that allows an adversary to achieve the goal of linking an individuals FIDO registrations.

In sections below, we provide detailed descriptions of the features that lead to the attack, then present an adversarial model and attack algorithm. Finally, we provide several examples of possible adversaries.

4.2.1 Remote CTAP Calls and Webauthn API Implementation

To describe how an adversary can measure execution time differences without user interaction we first have to explain two different types of user checks defined by the FIDO2 specification. The first one is *user presence* which requires the authenticator to check if a human actor is present to proceed with authentication. It is worth noting that popular implementations known from hardware tokens can be easily simulated (e.g., button click can be done by machine). The second check is called *user verification* and it aims to authorize the accepting party. There exists a variety of available implementations such as memorized secrets (e.g., PIN code) or biometric authentication (e.g. fingerprint).

User presence and user verification are configurable in the CTAP protocol as simple flags. User presence is always required during registration but the FIDO2 specification allows the CTAP client to modify both flags during authentication/assertion. This means that the CTAP client can trigger assertions without user input (also called

silent authentication). Silent authentication does not usually trigger any indication on the tokens themselves. E.g., the LED indicator that usually signals that the token is waiting for human action remains unchanged. It is easy to imagine a scenario where malicious software is probing a hardware token left connected to the users platform.

Silent authentication is a useful mechanism for FIDO2 clients, since they can use it to filter key handles in the *allowCredential* list provided by the WebAuthn API. The user is not bothered to provide a user presence check for each attempt and a CTAP client (e.g., a browser) can identify which key handle belongs to the authenticator. After finding the correct key handle the CTAP client continues with the assertion that requires user presence or verification. This functionality can be abused by sending an *allowCredential* list with key handles to check. In other words, an adversary can remotely trigger the execution of silent authentications on the token by creating a proper combination of key handles in the *allowCredential* parameter provided by the WebAuthn API.

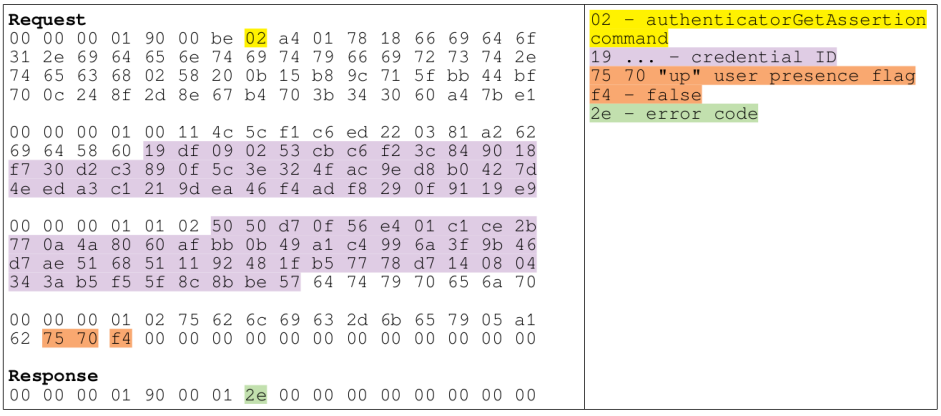


Figure 4.1: Data exchanged in a single CTAP silent authentication captured by intercepting USB traffic using Wireshark software triggered from Chrome browser

We found that all browsers considered in this study (Chrome, Brave, Firefox, Opera, Edge and Safari) parse an *allowCredential* parameter with multiple key handles in this way. Analysis of the Chromium browser source code (used in many commercial browsers, e.g., Chrome, Opera and Edge) revealed that silent authentication is executed whenever the *allowCredential* list contains more than one element (see [87] line

224-232). We further examined the data exchanged with a USB hardware token for the Firefox and Chrome browsers, which clearly showed silent authentications for each key handle in the *allowCredential* list are made until a key handle successfully authenticates, after which a non-silent authentication with the found key handle was triggered. The remaining browsers (Brave, Edge and Safari) perform a single user presence check when there are multiple entries in the *allowCredential* list, indicating that silent authentications are also used in this way. In Figure 4.1 we present data exchanged for a single silent authentication and highlighted the most important parts of the message.

The last obstacle the adversary needs to overcome is a user presence check of the correct assertion. The time of human action is not deterministic and it might require the user to find her hardware token, plug it in and execute the required action. An adversary can ensure the user is ready by executing the time measurement attack on the second consecutive WebAuthn assertion so that the user is already prepared for a user presence check. Further, to maximize the measurable time difference, the adversary can introduce multiple repetitions of the same key handle in the *allowCredential* parameter, spreading the user indeterminacy across multiple CTAP calls. We outline a second approach in Section 4.3.4 using the user's microphone to determine the point at which the user clicks the token's button and hence when key handle processing begins.

4.2.2 Difference in Key Handle Processing

FIDO authenticators use unique keys per relying party. This ensures the unlinkability of the user's accounts, however also introduces a storage problem since hardware tokens have limited memory and can only store a small number of on-device keys (also called resident keys). A common cryptographic technique used as a solution for this problem is key wrapping, i.e. storing an encrypted version of the secret key outside of the device.

Yubikey hardware tokens (manufactured by Yubico) are amongst the popular ones that use this technique [5]. Yubico's approach is to wrap the signing keys together with the corresponding origin/application ID or its hash value (to have a constant

[5]See e.g. how keys are stored in case of Yubico: https://developers.yubico.com/U2F/Protocol_details/Key_generation.html

size plaintext instead of a variable one). This ensures that the key handle can only be used with the correct relying party. It also protects against simple linking attacks where two malicious relying parties can try sending key handles from the other party to identify users. To protect the integrity of the plaintext authenticated encryption (AE) is used. The standard approach is to use the encrypt-then-mac approach, i.e. compute a message authentication code (MAC) on the ciphertext.

Key wrapping is usually done using a constant size master key. This is to limit the size of the key material stored which is also one of the goals of key wrapping. It follows that provided ciphertexts from different RP's the token will always correctly decrypt and verify the MAC. However, depending on the plaintext the actual execution can differ. In particular, after comparing the origin in the key handle with the one given as input, the token can either abort execution (in case of failure) or create an assertion (if the origin is accepted).

The simplest way one would implement this process is first to check the validity of the MAC and abort in case of failure, and then proceed with checking the origin. The former requires the computation of the hash value of the input origin. As noted above we have three cases: 1) abort on MAC verification, 2) abort on origin verification, and 3) complete execution. If the above implementation is used then there should be a time difference between cases 1) and 2).

We will now show that this is what probably happens for the hardware tokens for which we were able to prove the existence of a timing difference. It is worth noting, that without the firmware of the vulnerable tokens we are unable to pinpoint the actual reason for the difference.

To give an argument supporting our proposition let us take a look at the key wrapping decryption process implemented in Google's open-source OpenSK FIDO token implementation [88]. We show the interesting part of the OpenSK source in Figure B.3 (Appendix B.4). In particular lines 272 and 295. In the former, the token verifies the MAC for the key handle and aborts in case it is invalid. In the latter, the token compares the decrypted id of the relying party with the origin provided as part of the FIDO authentication data to ensure that the key handle is only used with the correct

relying party. Between those lines of code (lines 279-294) the token is doing other computations that, among others include AES CBC decryption of the key handle. It is easy to see that depending on the computational capabilities of the hardware this can lead to a noticeable time difference.

4.2.3 Adversarial Model

The goal of the adversary is to link FIDO2 registrations that were executed using the same authenticator. Under the assumption that the same authenticator is used by the same person, the adversary then has a connection between that user's accounts, and effectively create a user profile.

We assume the adversary has the capability to remotely control the flow of FIDO2 authentication, however the attack does not deviate from the FIDO2 specifications. Our adversary is an active attacker in the sense that they control JavaScript code executed on the victim's client and manipulate FIDO communications, however, the attack can only be performed during an authentication transaction initiated by the victim, and can only leverage valid modifications of FIDO2 messages without disrupting the protocol or deviating from the protocol definitions (workflow, syntax, validations).

To achieve these capabilities, the adversary either needs to be a trusted authentication provider or have the ability to inject malicious code and payloads into the authentication process (see Section 4.2.5 for examples). In this work we exclude adversaries that can execute CTAP calls directly (e.g., a compromised browser or malicious FIDO client). Adversaries with this stronger capability can directly execute silent authentications both to determine if key handles are present on the authenticator (user account linking) and to maliciously authenticate without user knowledge.
Below, we summarize the attacker capabilities:

Adversary can:

- execute and manipulate FIDO2 protocol messages,
- access key handles (owned or stolen) not presented by the victim,
- measure timing of WebAuthn authentication calls.

Adversary cannot:

- deviate from FIDO2 protocol specifications,
- trigger errors (as these would alert the victim).

4.2.4 The Attack Concept

First, notice that the adversary described above is in possession of data that includes key handles corresponding to the authenticators of users. For simplicity we focus on a single attack scenario: an adversary that implements service $\mathcal{A}$ and tries to distinguish if the key handle $\mathsf{key\_handle}_{\mathcal{B}}$ for service $\mathcal{B}$ corresponds to the user authenticating with handle $\mathsf{key\_handle}_{\mathcal{A}}$. The malicious queries can be build out of $\mathsf{key\_handle}_{\mathcal{A}}$, $\mathsf{key\_handle}_{\mathcal{B}}$ and random handles $\mathsf{key\_handle}_{\mathcal{R}}$. We use random key handles as a proxy for key handles generated by different authenticators. The attack is successful assuming there is a noticeable difference in the time it takes the authenticator to process key handle $\mathsf{key\_handle}_{\mathcal{B}}$ and $\mathsf{key\_handle}_{\mathcal{R}}$ when connecting to service $\mathcal{A}$. We show in Section 4.3.2 that this assumption is actually true for some existing hardware tokens and in Section 4.2.2 we discuss potential reasons for this time difference.

The first step of the attack is to find the baseline execution time t_e, which corresponds to the time it takes for an authenticator to answer a WebAuthn API call with n random key handles and one valid key handle for service $\mathcal{A}$ plus the time for the non-deterministic user factor. The key handle list is placed in the *allowCredential* field of the call (see Figure 4.2). To measure the time the adversary uses timers to enclose the navigator.credentials.get call to the browsers WebAuthn API. The second part of the attack is performed in a similar manner, however this time n copies of $\mathsf{key\_handle}_{\mathcal{B}}$ are used in place of random key handles. The resulting time t_d is then compared with t_e. In case of a noticeable difference the adversary concludes that $\mathsf{key\_handle}_{\mathcal{B}}$ is also registered in the authenticator, and the user has an account with service $\mathcal{B}$. Note that once t_e is know the adversary can omit the first step of the attack (assuming the same user for $\mathcal{A}$ is connecting). Algorithm 1 provides an overview of the attack.

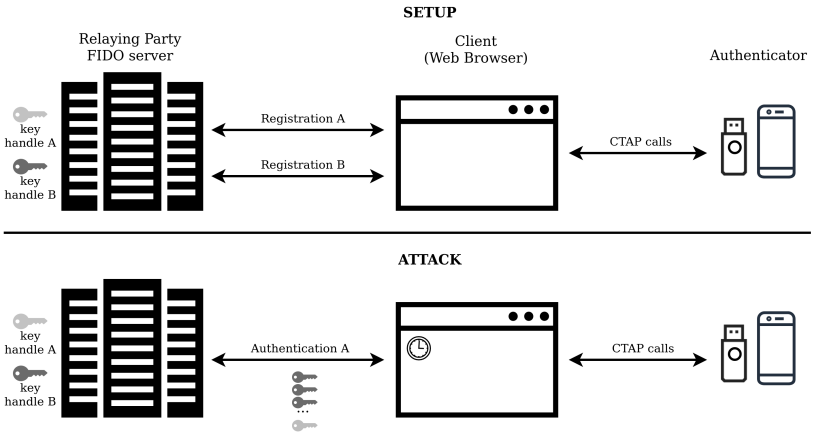


Figure 4.2: Setup and attack phases for FIDO2 side-channel attack. During authentication to $\mathcal{A}$, the attacker builds an *allowCredential* list with multiple $\mathsf{key\_handle}_{\mathcal{B}}$ and $\mathsf{key\_handle}_{\mathcal{A}}$.

The value n (the number of requested silent authentications) is an attack parameter and depends on the attacked platform. The higher it is the less the non-deterministic user factor influences the attack, since by replicating the key handle in question (e.g. $\mathsf{key\_handle}_{\mathcal{B}}$) it is divided across all executions.

4.2.5 Possible Adversaries

We present five possible adversary types: three that are providers of FIDO2 authentication, one that is capable of intercepting and manipulating FIDO2 and client side JavaScript, and one that uses injected JavaScript code to trigger the attack.

All adversaries execute the same attack concept, however details of the attack setup differ. The two *allowCredential* lists (one containing random key handles, the other a candidate key handle — Algorithm 1 steps 2,3) may be created on the FIDO server (as shown in Figure 4.3) and communicated to the client via WebAuthn transactions, may be created in a malicious server and inserted into WebAuthn transactions, or may be created by attacker JavaScript code on the client. In all cases JavaScript code timing the execution of WebAuthn calls is then executed on the client (Algorithm 1 step 4), the user performs user presence checks (step 5) and results sent to the attackers server (steps 6,7). Schematics of these additional attack scenarios are illustrated and described in Appendix B.3.

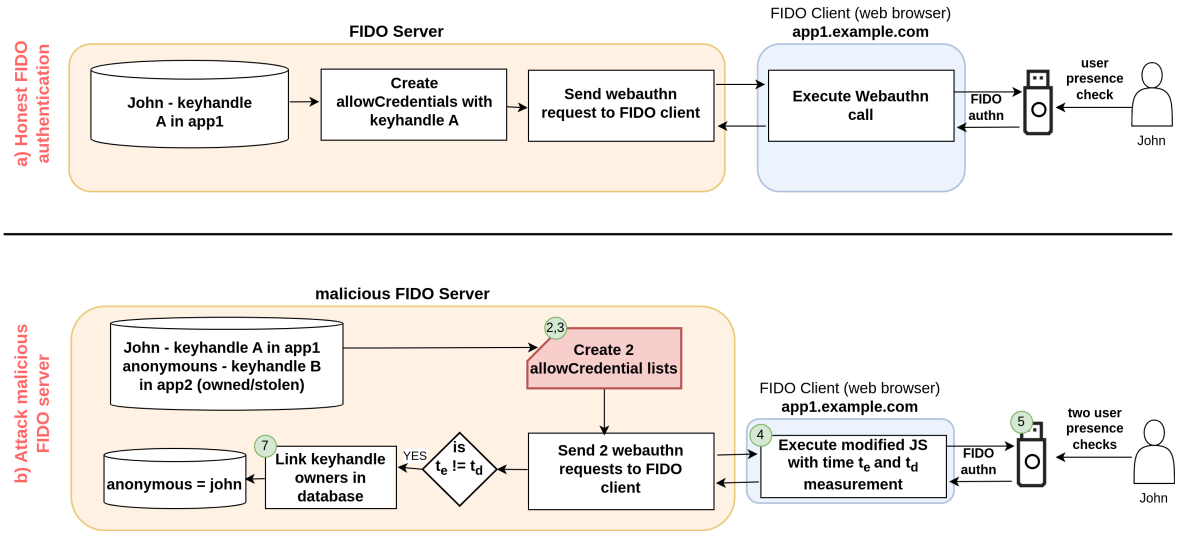


Figure 4.3: FIDO2 timing attack example diagram. Numbers in green circles correspond to steps in Algorithm 1. More attack examples can be found in Appendix B.3.
a) The first diagram illustrates an honest FIDO transaction.
b) The second diagram presents an attack flow triggered by a malicious FIDO server.

Algorithm 1 High level attack algorithm

1. *Victim* : starts FIDO2 transaction
2. *Adversary* : prepares *allowCredential* list with n random key_handles and $\text{key\_handle}_{\mathcal{A}}$ (can be omitted if t_e known)
3. *Adversary* : prepares *allowCredential* list with n copies of $\text{key\_handle}_{\mathcal{B}}$ and one $\text{key\_handle}_{\mathcal{A}}$
4. *Adversary* : forces FIDO client to perform WebAuthn calls with prepared *allowCredential* lists and measures times of execution (t_e and t_d)
5. *Victim* : performs user presence checks
6. *Adversary* : compares measured times (t_e and t_d)
7. *Adversary* : Links identities if times do not match

The first adversary type provides FIDO services for a single application where users can benefit from multiple accounts. An example of this use case is a cryptocurrency exchange (e.g., one of the biggest cryptocurrency exchanges, Binance[6], implements FIDO2 as a second factor). Having multiple accounts on the exchange can reduce the traceability of one's transactions, hence keeping them unlinked is of great value for the user.

[6]https://www.binance.com/en

The second consists of more complex applications that provide not only core services but can also act as an identity provider, allowing users to login to third party applications using their accounts and the login flow of the identity provider. Examples are services similar to Google which provide a "Login with XYZ" service. In this setup, we can imagine users with two Google accounts that do not want associated third party accounts to be linked.

The next group offer FIDO2 as a service for third parties. They make it easy to integrate strong authentication but it also means that FIDO2 related data is kept on their servers and they execute FIDO2 authentication flows. An example provider of such a service is Duo Security[7]. Such service providers are in possession of data from different services and hence are capable of executing cross service linking.

Our last two possible adversaries are not owners of FIDO2 data, but can obtain either stolen data or spy on user authentications. First, any service acting as an SSL termination proxy can intercept and modify FIDO2 message payloads and modifying FIDO related javascript, and are thus capable of sending malicious payloads and executing timing code on the client. A commercial example of such a service is Cloudflare[8]. Similarly to the FIDO2 as a service case, the proxy can gather FIDO2 data from a diversity of applications.

Finally, any actor that is able to modify JavaScript code is in a position to execute our timing attack. Considering that the JavaScript XSS (Cross Site Scripting) attack vector has remained in the OWASP TOP 10 list of vulnerabilities for many years [19], we consider this variation of the attack as highly probable. Similarly to the proxy example, FIDO2 data can be gathered from user authentications in compromised browsers or obtained from data breaches. Note that FIDO2 credentials *cannot* be stolen in this way.

We note that in all cases, stolen FIDO2 data (in the form of key handles) can be utilised by attackers to broaden the attack scope. Additionally, adversaries can use context metadata (e.g. account data linked to key handle) to narrow down the set of

[7] https://duo.com
[8] https://www.cloudflare.com

potential key handles to check.

Considering the number of possible adversary types, we believe that the attack described below has a high potential to be deployed in the real world and can violate the privacy of users secured with FIDO2.

4.3 Results

In this section we present the methodology and results of our tests. FIDO2 with WebAuthn is designed for web applications and it specifies a well-defined path between the relying party, browser, and authenticator. Our attack faithfully follows the FIDO2 flow, which executes through FIDO clients and authenticators. For this reason we focus our analysis on those two elements. Our test sessions were recorded and are available online together with source code[9].

4.3.1 Methodology

The methodology of our test suite includes two parts. Firstly, we measure silent authentication directly on the authenticators to identify vulnerable devices. In the second part, we executed remote timing measurements on the WebAuthn API using the vulnerable devices from the first phase.

In the first phase, our goal was to measure silent authentication directly on the FIDO2 authenticators. For the USB hardware authenticators we used the open source Yubico FIDO2 library[10] to make CTAP calls directly. Unfortunately, the same is not possible on the Android platform because the available SDK does not implement direct access to CTAP and the WebAuthn API forces a user presence check. In the case of iOS, access to both WebAuthn and CTAP are unavailable, and we were unable easily to test for time differences.

We measured the time between request and response for multiple independent silent authentication calls with a single key handle in the *allowCredential* list. We used either

[9]Test recordings: https://osf.io/t7dpa/?view_only=c8595da6c6d34fadb87f2f6db7e5d626
[10]https://developers.yubico.com/libfido2

a random key handle (with the correct length) or a correct key handle with an incorrect origin value. When there was an appreciable difference, we identified the authenticator as vulnerable.

For the second phase we built a Relying Party in Node.js which serves a test HTML page with WebAuthn API executions in JavaScript and measures execution times. We deployed the software on Amazon AWS with a public IP address which we accessed via several web browsing platforms to obtain authentication timing data.

4.3.2 FIDO2 Hardware Authenticators

The number of authenticators available on the market is constantly growing. Because FIDO2 is an open source specification, there is no public record of commercial FIDO2 authenticators. However, an estimate can be made based on FIDO Alliance voluntary certification, which at the time of this writing holds 140 certified FIDO2 authenticators. The list contains platform, roaming, hardware and software authenticators. Notably, our attack can be launched against any type of FIDO authenticator. We chose to evaluate the most secure and numerous (111 certified devices) FIDO authenticator type: hardware roaming authenticators.

We selected eight hardware authenticators that are certified by the FIDO Alliance at level 1 [68] with an aim to provide a representative view of the available options. Our selection provides a broad range of price ranges, vendor sizes, features, and countries. We picked "Yubico Yubikey 5 FIDO2 USB-A"[11], "HyperFIDO Titanium Pro"[12], "Google Titan"[13], "Token2 T2F2 Bio"[14], "Feitian K26"[15], "TrustKey G320H"[16], "Kensington Verimark Guard"[17], and "AuthenTrend ATKey.Pro"[18].

While examining the Yubikey token, we observed a defense mechanism: After 10

[11] https://www.yubico.com/products/yubikey-5-overview
[12] https://www.hypersecu.com/products
[13] https://store.google.com/us/product/titan_security_key
[14] https://www.token2.net/shop/product/token2-t2f2-bio-fido2-u2f-and-totp-security-key-with-fingerprint-protection
[15] https://www.ftsafe.com/products/FIDO/BIO
[16] https://www.trustkeysolutions.com/en/sub/product.form
[17] https://www.kensingtonstore.com.au/products/verimark-guard-usb-c-fingerprint-key-fido2-webauthn-ctap2-fido-u2f-cross-platform
[18] https://authentrend.com/atkey-pro

incorrect attempts a randomised delay is added, which prevents our attack from succeeding. Google Titan, Token2 T2F2, TrustKey G320H, Kensington Verimark Guard, and AuthenTrend ATKey.Pro authenticatorGetAssertion times were not distinguishable. We successfully executed timing attacks on HyperFIDO Titanium Pro (average difference of 10ms per execution) and Feitian K26 token (average difference of 2ms per execution) — see Table 4.1 for an overview and Figure 4.4, left panes for timing results for vulnerable tokens (see Appendix B.2 for other tokens).

Table 4.1: Test results indicating hardware tokens vulnerable to timing attack.

	Yubikey 5	Hyperfido Titanium Pro	Google Titan	Token2 T2F2-Bio	Feitian K26	TrustKey G320H	AuthenTrend ATKey.Pro	Kensington Verimark Guard
Is vulnerable	✗	✓ (10.07)*	✗	✗	✓ (2.21)*	✗	✗	✗
Is upgradeable	✗	✗	✗	✗	✗	✗	✗	✗

* Average difference (ms) for silent authentication between random and bad origin key handles

A simple threshold based classifier correctly identifies key handles with a 0.1% error (HyperFido) and 6% error (Feitian) if user presence timing is known (for example, using an audio signal as described in Section 4.3.4). We further simulated noise from user presence checks using results from a small user study (see Section 4.3.4) by adding a randomly sampled user presence check timing result to each CTAP timing result. These adjusted CTAP timing figures (one set of figures for each subject in the user study) were then used to determine a threshold as before and to predict whether each CTAP call contained key handles present on the token (Figure 4.4, right panes). In all cases, 70% of the CTAP timing data was used to determine the threshold and the remaining 30% to evaluate the resulting classifier. Note that an attacker may combine estimates from multiple user authentication sessions to further mitigate noise from user presence checks.

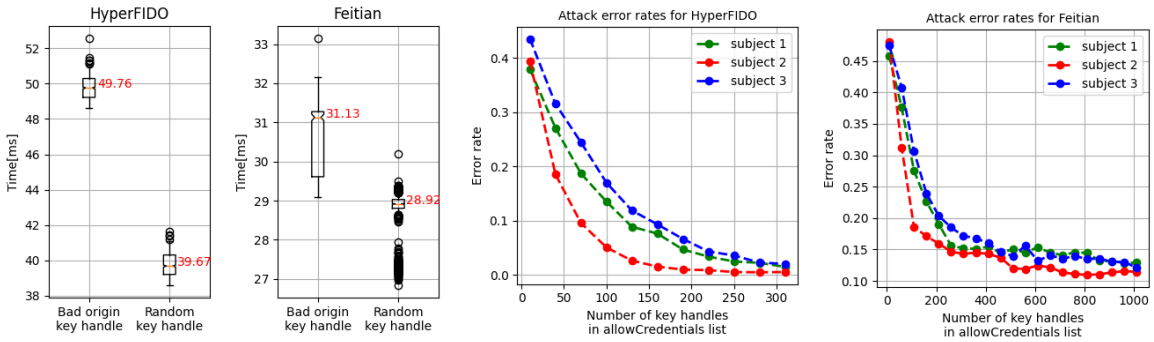


Figure 4.4: Measurements of response times of vulnerable tokens HyperFIDO Titanum PRO and Feitian (left). Right: Error rate for a simple threshold classifier with user presence noise from the user study.

4.3.3 FIDO2 Clients

The second element in the FIDO2 flow that can be vulnerable to timing measurements of assertions is the FIDO2 client. The most popular FIDO2 clients are web browsers. If the WebAuthn API is supported by the browser, each execution of the WebAuthn API in JavaScript is translated into CTAP calls to the FIDO2 authenticator.

Table 4.2: Test results indicating if browsers execute silent authentications for all key handles in *allowCredential* list, thus allowing timing attacks when combined with a vulnerable authenticator.

	Chrome	Brave	Firefox	Opera	Edge	Safari
Windows 10*	✗	✗	✗	✗	✗	N/A
MacOS 11.2.3	✓	✓	✓	✓	✓	✓
Ubuntu 18.04	✓	✓	✓	✓	✓	N/A
Android 10	✓	N/S	N/S	✓	N/S	N/A
iOS 14.4.2	N/S	- †	N/S	N/S	N/S	✗‡

✓, ✗- Allows / does not allow attack
N/S - FIDO2 not supported
* Browsers use native Microsoft WebAuthn API
† Brave browser for iOS has a custom implementation with hardware tokens only
‡ Safari uses native iOS WebAuthn API

We tested six popular web browsers (we included the "Brave" browser as the one that focuses on privacy) running on 5 widely used operating systems for desktop and mobile (see Table 4.2). We did not evaluated Internet Explorer because it does not implement WebAuthn. We used the latest versions of browsers as of 6th of April 2021.

Four of the six tested browsers are based on the Chromium engine (only Firefox and Safari have completely independent source code). We used a HyperFIDO Titanium Pro hardware token because we knew that it is vulnerable to timing attacks (Section 4.3.2).

On the Windows platform (Windows 10 Home 19042.867) all supported browsers passed control to the Windows WebAuthn API. We were unable to execute the attack with a large number of key handles. The Windows WebAuthn API introduces a limit on how many requests can be sent to the token. Empirically, using Wireshark USB logs we confirmed that 20 silent authentication attempts are made before failure.

On MacOS Big Sur (Version 11.2.3 on MacBook Pro), all tested browsers showed vulnerability for our timing measurement. We experienced unexpected behavior from the Safari browser: Test attempts with 64 or more key handles in the *allowCredential* list cause Safari to crash, hence our attack was limited to only 63 key handles. Though this would reduce the efficiency of our attack, we recognize it as a bug and not a security feature, thus we conclude that Safari is vulnerable.

The last desktop platform which we tested is Ubuntu 18.04 (one of the most popular desktop operating system from the Linux family) for which we were able to successfully execute timing measurements on all browsers.

We tested Android 10 as it is the most popular Android OS version at the time of this writing. The test was executed on five phones: Google Pixel2, Samsung A2, Mi8, Motorola One Vision and Oppo Reno2 Z. All tested phones are equipped with a fingerprint scanner. We found that only Chrome and Opera support WebAuthn executions. Similarly to Windows, WebAuthn control is given to the Android system where the user can select which token type should be used. We tested a native Android authenticator secured with fingerprint (the "Use this device with screen lock" option). We did not observe a timing difference during our attack.

The iOS platform has the most limited group of browsers supporting FIDO2 as Apple has not yet released a native API for WebAuthn. In our tests (executed on iOS 14.4.2), only Safari and Brave allowed WebAuthn calls. Safari uses iOS native APIs which allow using iOS authentication mechanisms (e.g., TouchID). We found that the iOS API works as a client with client-side storage and because we couldn't see any

difference in response times, we suspect that the *allowCredential* list is filtered with key handles saved on the client-side. The Brave browser uses a custom implementation to connect to a hardware FIDO2 token. We were unable to check our physical tokens on iOS because of the incompatibility of the iOS lightning port with our tokens.

4.3.4 Dealing with User Presence Checks

We have seen that using multiple key handles in a single CTAP call can reduce the impact of the indeterminacy of the time taken by the user to perform the user presence check. In this section we present two additional approaches to reduce its impact and a small pilot study to quantify that indeterminacy.

Priming User Presence Checks

Table 4.3: Timing variation results for FIDO2 authentication time from user study.

	1st authn		**2nd (primed) authn**	
Subject	**Mean**	**Std. Dev.**	**Mean**	**Std. Dev.**
1	5041 ms	943 ms	750 ms	227 ms
2	3980 ms	585 ms	344 ms	116 ms
3	5441 ms	844 ms	707 ms	277 ms

The first strategy to reduce the impact of user presence checks is to prime the user by requiring them to perform the check twice: the first time intended for them to find the token and insert it, the second for timing. The user would be told that there was a problem with authentication and that they need to repeat it. Our hypothesis is that this approach would lead to substantially more consistent timing for the second check.

To verify this hypothesis we performed a small proof of concept study among authors that simulated this sequence of events. We built a simple web page that requests FIDO authentication. Our subjects were instructed to insert a FIDO2 hardware token once the browser shows the authentication prompt and take it out once authentication is successful. We notified them that authentication might not work first time, and in that case the token doesn't need to be removed between attempts. The authentication was repeated 50 times for each subject. Each authentication was triggered after

a randomized time interval to limit preparedness and thus minimise bias. The results show that the second consecutive authentication takes far less time and has far less variability as we hypothesised (see Table 4.3). All study participants were authors, and thus the study was exempt from ethics review (confirmed by the Macquarie University Ethics Committee). We acknowledge that, despite our best intentions and measures to minimise bias, the study was in the end conducted by authors and bias may remain.

Alternative Time Measurement

In our attack the adversary builds an *allowCredential* list with multiple instances of the same key handle to limit the influence of user action. Some configurations are resistant to this kind of attack by limiting the number of allowed entries in the *allowCredential* list (see Section 4.3). To circumvent that limitation we propose an alternative method of measuring the execution time in which the measurement starts immediately after the user presence check. This way we eliminate the non-deterministic delay and can use smaller sized *allowCredential* lists.

The authenticatorGetAssertion WebAuthn API call implemented by browsers enforces a user presence check which is not uniformly implemented in different FIDO2 authenticators. This check requirement introduced additional elements in the manufacturing process of hardware tokens. The factors that influence the decision on what kind of interface is selected include not only security but also production cost and user experience. For example, some Yubico tokens use capacitive touch sensors whereas HyperFido tokens use a physical button, which is one of the most popular solutions. Inspired by the work in the area of acoustic side-channel attacks (e.g. Genkin at al. [82]) we observed that physical buttons on authenticator tokens emit a characteristic sound when used. We use this observation in our modified attack described below.

The Modified Attack

In this variation of the attack, the service does not have to send multiple keys, however it needs to record audio using the attacked platforms microphone. This can be achieved

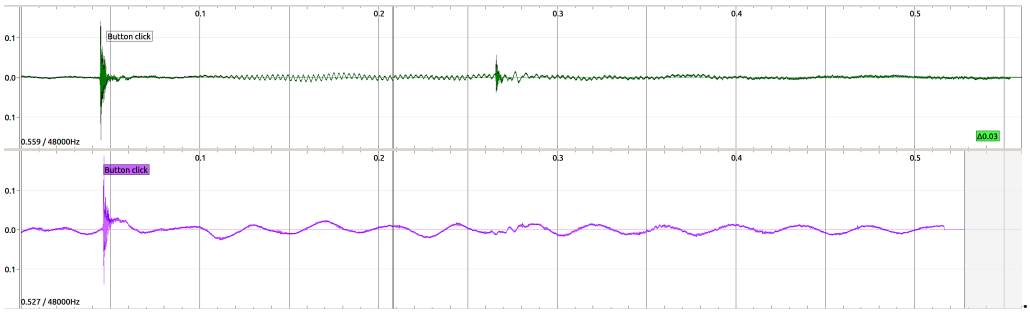


Figure 4.5: Audio recording of FIDO assertion on HyperFido Titanium Pro. Green: an attempt with a bad origin key handle. Pink: an attempt with a random key handle. The initial silence is truncated to allow comparison.

using the MediaStream API [11], which allows the capture of sound directly to a JavaScript object. We wrapped the execution of *navigator.credentials.get* with sound recording code. After the recording is finished, the sample is sent to the backend part of the attacker/service for processing. Figure 4.5 presents recordings from two attempts, using a key handle with bad origin and using a random key handle. The test was performed in a home environment on a Dell XPS 15 9570 laptop with HyperFido Titan Pro token. The button click action is easily distinguishable and the time difference between attempts simplifies the identification of the key handle with bad origin.

It is easy to see that this attack requires a strong adversary that is granted access to the microphone through the MediaStream API. It also requires that the time difference between executions of random and bad origin key handles are high enough to be distinguishable in the recording, and the token needs to be constructed with a button that generates noticeable sound. Many services use the MediaStream API to provide video-conferencing features and once consent is given, the application can trigger recordings freely and would be in position to perform the attack. Interestingly, the microphone usage consent window is presented to the user as a standard browser dialogue window which appears with the same location and "look and feel" as WebAuthn dialogues, and is shown just before the WebAuthn window, so it could be easily accepted by mistake, enabling the attack in other cases.

4.3.5 User Experience

Our attack has minimal influence on how FIDO2 authentication is perceived from a user perspective. Authentication proceeds as normal due to the correct key handle at the end of the list, but with a slight delay. The time difference can be noticeable but it is indistinguishable from network or browser slowdown. Therefore, we claim that it is unlikely for the user to notice any irregularities in the authentication process. Examples of user experience can be observed in attached recordings[19].

4.3.6 Vulnerable FIDO2 Deployments

In this section we discuss how FIDO2 is deployed in publicly available web applications and how this relates to our proposed attack. In terms of production applications, security of the solution is not the only aspect to be considered. For example, introduction of an additional factor for authentication brings additional cost and potential disruption to the business. Perhaps the most surprising aspect of the FIDO2 environment is user adoption and experience. Based on previous research concerning the usability of FIDO2 hardware tokens [83][50], it is evident that the shift toward more secure authentication methods encounters challenges rooted in human behavior.

Our proposed timing attack is based on a non-resident key scenario, which we believe covers the majority of publicly available FIDO2 implementations. To verify the broad applicability of our attack on FIDO2 deployments in the wild, we employed a web crawler designed to identify FIDO2 authentication on public websites and record the WebAuthn configuration (i.e., whether non-resident keys are allowed). We checked the 1 million most popular DNS records from the Cisco Umbrella set[20] for the presence of WebAuthn protocol executions between 10 December 2020 and 20 December 2020. Because WebAuthn executions can be found in the javascript resources of web applications we targeted our crawler to check for the presence of the *navigator.credentials.create* property with *public-key* type. We acknowledge that some applications have more complex authentication procedures that do not reveal usage of

[19]Test recordings: https://osf.io/t7dpa/?view_only=c8595da6c6d34fadb87f2f6db7e5d626
[20]https://umbrella.cisco.com/

the WebAuthn protocol (e.g., WebAuthn is dynamically loaded after completing first factor of authentication). Fortunately for us, initial authentication (e.g., username/-password) implies that FIDO2, if used, will be configured as a second factor without resident keys in these cases.

The results confirmed our belief about FIDO2 usage in the wild. We gathered 684 records of WebAuthn executions from which we extracted following groups. Most findings (52%) came from applications that reuse open-source tools (e.g., the discord platform[21] is frequently used as community forum, nextcloud[22] is used as a file storage and share service). In those cases FIDO2 is implemented with non-resident keys (as second factor authentication). The second identified group (16%) consists of federated authentication platforms (e.g., wordpress.com, github.com, Microsoft login). In this group only Microsoft login with passwordless FIDO2 used resident keys. In the last group, we gathered applications that use self-implemented FIDO2 authentication, from which all were configured to use FIDO2 with non-resident keys (as a second factor).

Our final conclusion from the crawling exercise is that in the public space FIDO2 is mostly used as a second factor mechanism with non-resident keys. Passwordless authentication (with resident keys) is in the early adoption phase and only a few providers give this option. In terms of our timing attack, this means that the majority of Relying Parties in the wild are vulnerable to and have the potential to launch our attack to link key handles.

4.4 Discussion

Here, we discuss the features that were key enablers of the attack presented in this chapter. We hope that our findings will contribute to enhanced security of all certified FIDO2 tokens.

Firstly, the silent authentication mechanism opens a path to bypass the human factor in FIDO2 authentication. We acknowledge that it simplifies the automation of the pre-authentication processes, nevertheless it introduces threats that, from a privacy

[21] https://discord.com/
[22] https://nextcloud.com/

perspective, may outweigh its benefits. We believe that FIDO client's implementations should introduce additional safeguards on the silent authentication process (as described in Section 4.4.1). Moreover, we observed that rate limiting techniques (e.g., adding delay after a number of unsuccessful calls) are not popular in FIDO2 authenticators. This might allow for enumeration, brute-force or as in our case timing attacks.

Additionally, we want to emphasize the importance of FIDO Alliance certification. We acknowledge the utility of graded certification levels, however understanding differences between certification levels requires expert knowledge not available to a regular user. Our attack showed that authenticators with L1 certification cannot guarantee all FIDO2 security and privacy goals. Therefore, we believe that all authenticators should be evaluated against L3 controls, which guarantee on-device testing.

4.4.1 Attack Mitigation

The vulnerabilities responsible for our timing attack occur on two loosely coupled layers of FIDO2 transactions: FIDO clients and FIDO authenticators. Considering the number of authenticators and clients available on the market and already deployed, a complete mitigation solution has to address both layers.

We present four mitigation strategies, two that apply to authenticators, one that circumvents our attack (at a cost) via FIDO configuration and one that applies to FIDO clients (e.g., browsers).

Via Constant Time Execution in Authenticators

The easiest way to protect against our attacks would be to update the firmware of hardware tokens and implement the execution in a way that there is no time difference between checking random key handles and key handles for different origins. Below, we demonstrate how this can be achieved using a technique that has already been implemented in some hardware tokens. Unfortunately, this only mitigates the problem in case of new users that buy a hardware token with updated firmware, as the majority of hardware tokens do not allow firmware updates. Therefore, we provide two more

mitigation strategies that can be easily implemented in FIDO servers and clients.

Via Key Derivation Function in Authenticators

An alternative way to generate a keypair is to use a key derivation function keyed with a master secret and seeded with data related to the relying party. This process requires the authenticator to only store one master secret key (i.e. AES key) which is then used to pseudorandomly derive the signing key for the FIDO authentication process.

This is a well-known technique and has already been implemented by some tokens. However, due to their closed firmware, we are unable to verify how many tokens implement key generation in that way. A notable exception is the SoloKey FIDO token which comes with an open-source firmware [172] for which we show the key generation function (see Figure B.4 Appendix B.4). The implementation also uses key handles which in this case are just random values. During the authentication process, this random value is the used as the `data` argument for the key generation function (`data2` is left empty).

In this approach the key handles are just random values, hence the authenticator's processing time for every key handle is the same and the authenticator is not vulnerable to our attack.

Via Resident Keys (FIDO Configuration)

Our attack utilises a weakness in incorrect implementations of key handling (i.e., handling non-resident keys) in authenticators. Methods such as key wrapping were introduced to solve a memory problem on authenticators, which need to store unique signing keys for each relying party (for privacy reasons). Alternatively, the FIDO2 protocol can be configured to store keys directly on the authenticator (residence keys), which eliminates our attack vector. Even though, the introduction of resident keys on the FIDO server might seem trivial (setting *requireResidentKey* flag in the registration request), the consequences for FIDO authenticator users are significant. Firstly, not all FIDO authenticators support resident keys. Moreover, the modification of key storage technique in an existing authentication system requires all users to perform the FIDO

registration process once again. Finally, the storage capacity of key handles in roaming authenticators is limited (e.g., Yubico keys allow up to 25 keys), which noticeably reduces their utility. We found only one deployment of FIDO in the wild that uses resident keys (Microsoft AzureAD passwordless login). All other implementations we found in the wild (Section 4.3.6) use non-resident keys.

Via Client Update.

The FIDO client attack vector can be mitigated by changing the way browsers and other clients implement the WebAuth API *allowCredential* parameter. In particular, we propose the following mechanisms.

1) Deduplication of the *allowCredential* list before making CTAP calls, which would remove all repetitions of bad origin key handles and thus prevent amplification of the time difference.[1]

2) Silent authentication errors can be delayed by a random value that is large enough to render the attack ineffective due to a high error rates.

3) The size of *allowCredential* can be limited to e.g. 10 or 20 elements. This should still preserve the functionality since most users will never register more than 10 tokens with a single relying party but users remain vulnerable to the second (weaker) form of attack using audio detection of user presence checks. Note that this is already the case with the Windows 10 WebAuthn implementation, which limits to 20 elements.

Note that these approaches are hardware-token-independent, and as such, a scaled mitigation of our attack can be achieved through a mandatory software upgrade (e.g., an automatic browser update).

4.5 Conclusion

In this chapter, we introduced a conceptual attack against the privacy of the FIDO authentication process. At first glance the attack is based on strong assumptions

[1]The Chromium team acknowledged our finding as an information disclosure vulnerability and suggested deduplication as a mitigation.

about the capabilities of the adversary. However, we demonstrate that the chain of flows in protocol and existing implementations allows a remote adversary to break the unlinkability of FIDO2. We built a proof-of-concept and showed that the attack is possible for many configurations of FIDO clients and authenticators and showed that the majority of FIDO2 providers in the wild use non-resident key handles and are thus susceptible to accounts with them being linked to other services by malicious actors.

In the course of our research we were not able to investigate all available authenticators, however, based on the fact that the vulnerable authenticators we found are manufactured by major security vendors, we expect this flaw to be present in other products as well. Even more concerning, simply identifying and mitigating the vulnerability in hardware tokens may not suffice, as the majority of hardware tokens lack support for firmware updates. We proposed a mitigation mechanism that involves the client side (i.e. browsers) and is independent of the authenticator which prevents remote attacks via web pages. This approach allows the user to update their FIDO client (typically a browser) and still use vulnerable tokens safely for web based authentication via the browser.

The proposed mitigations for our attack have been implemented and publicly released, improving the privacy of FIDO2. In the context of our proposed system, addressing this issue in public FIDO2 clients and reviewing privacy mechanisms in FIDO2 has provided us with a solid foundation to design a system that enhances FIDO2 privacy guarantees, as presented in the next chapter.

5

Fast IDentity Online with Anonymous Credentials (FIDO-AC)

This chapter is adapted from the work titled "Fast IDentity Online with Anonymous Credentials (FIDO-AC)" published in the 32nd USENIX Security Symposium (USENIX 2023). Authors: W. Yeoh, M. Kepkowski*, G. Heide, M. A. Kaafar, L. Hanzlik*

In the preceding chapters, we have conducted an in-depth analysis of FIDO2, examining its integration usability, and privacy. Our investigation has revealed that FIDO2 is a highly suitable candidate for our requirements to build a privacy-preserving system for authentication and authorization. However, we also learned that authentication and authorization are coupled and usually consecutive processes. From an access management perspective, they determine the resources or services that can be provided

*Equal contribution.

to the requester. Unfortunately, although authorization follows authentication, these processes are typically not linked. These disjointed processes often lead to a situation where authorization attributes (e.g., age) and the authentication method (e.g., username and password) can come from different identities. Additionally, authorization attributes (except environments with heightened identity assurance requirements) are usually not verified when collected, relying solely on the honesty of users without any trust anchor.

The collection of authorization attributes, especially if they are verified (e.g., based on passport data), poses a significant leakage risk to the organization's reputation and the privacy of users. What is worse, once breached, authorization attributes that represent Personally Identifiable Information (PII) such as date of birth cannot be changed. Interestingly, for authentication and authorization use cases, only the decision (e.g., allow/deny) is required. Thus, the collection of PII data is only necessary due to the incorporated solutions.

In this chapter, we introduce a FIDO-AC system, strategically devised to confront the previously mentioned challenges through the utilization of privacy-preserving technologies. Rooted in the foundation of the FIDO2 protocol, FIDO-AC seamlessly incorporates anonymous credentials, which are issued by reputable entities, including governmental bodies. Our proposition encompasses a readily deployable solution tailored for industrial adoption, capable of seamless integration with existing deployments. Furthermore, we provide a functional prototype alongside a comprehensive evaluation.

5.1 Introduction

Web authentication is a crucial component of the digital world and the Internet we know today. The predominant web authentication method is via the login and password mechanism. The password is considered a single factor of authentication. In most modern applications, users are recommended to use multiple authentication factors. Such a factor could be an SMS code sent to the user's phone number or a designated mobile application requesting the user's acknowledgment.

The state-of-the-art solution is, however, based on cryptographic tokens. Those can store cryptographic keys and perform public key cryptography. The tokens, introduced by the Fast IDentity Online (FIDO) Alliance, are the most prominent instantiation of this idea, and together with the open-source FIDO2 protocol are a strong candidate for building advanced authentication frameworks. However, the main disadvantage of the current solution is that there is no link between the user's attributes and the authentication process, which limits the potential application space or forces the service to use ad-hoc solutions that are not bound to FIDO authentication.

The importance of attribute-based authentication is known in the research community, which has proposed many solutions. The most interesting ones are anonymous credentials, which allow users to disclose the attributes to service providers arbitrarily. Unfortunately, as time has shown, many of these approaches are not used in the real world and are far from what we consider practical. No tools and methodologies exist to efficiently combine anonymous credentials and attributes, in general, with the FIDO authentication process. The Meta Research group (formerly Facebook Research) reached the same conclusion . They issued a call for projects to develop such solutions [*]. Potential solutions to this problem would significantly influence how authentication systems are used and what we can use them for. One of the use cases is age verification, which is not a problem in most cases but becomes one if the service is legally obligated to check age (e.g., for selling alcohol or serving adult-only content). Even though those websites are required to verify the user's age, in practice, they ask the user to assert without further verification.

Those solutions are not limited to age verification but involve many practical systems where data minimization is needed but not implemented. This is evident due to many data breaches leaking the full data of identity documents. An interesting example is the 2022 data breach suffered by the Australian telecommunication company Singtel Optus [29], in which the hackers gained unauthorized access to two unique identity documents. The scope of this attack was significant because Singtel Optus did not

[*] https://research.facebook.com/research-awards/2022-privacy-enhancing-technologies-request-for-proposals/

employ techniques to minimize the personal data stored in the database, such as using attribute-based authentication that would allow the user to disclose only the minimal data required for identity check.

Contributions. This chapter introduces a novel approach for presenting claims in a privacy-preserving manner through a commercially recognized authentication protocol. To the best of our knowledge, our design is the first that presents an innovative and generic way to combine a privacy-enhancing technology (e.g., anonymous credentials and eID solutions) with a strong authentication protocol (i.e., FIDO2). Our approach to the framework definition is as follows. We introduce the building blocks of FIDO-AC and define the requirements for the system. Then, we formally define the notion of passwordless authentication with attributes, which we later extend with a mediator party to meet our requirements. Our security models can be of independent interest and used as a foundation for securely integrating attributes into the FIDO standard. In the next step, we introduce the system design and formal protocol flow, which we enrich with a security and threat analysis. Finally, we present our implementation of the FIDO-AC system and its performance evaluation. Notably, the design of our system recognizes and addresses the well-known challenges in integrating the existing deployments. Therefore, we believe that FIDO-AC can be effortlessly used in any commercial solution to elevate the privacy of Personal Identifiable Information (PII). To summarize, our contributions are as follows:

- **FIDO-AC system.** We propose a complete and industry-ready solution for utilizing anonymous credentials with local or remote attestation through a FIDO2 channel.
- **FIDO2 Extension.** We introduce a new FIDO2 extension and a mechanism to bind the extension data with the FIDO2 assertion in the constrained environment of the WebAuthn API implementations.
- **System evaluation.** We provide a comprehensive evaluation of the FIDO-AC framework. We discuss the security and privacy properties, as well as the usability from the user's and relying party's points of view.
- **Implementation.** We prove the feasibility of our design by developing and openly

publishing a prototype implementation.

5.2 FIDO-AC Technologies

This section will briefly explain electronic identity documents (focusing on ePassport) and anonymous credentials. The description of the FIDO2 protocol can be found in Chapter 2.

5.2.1 Electronic Identity Documents

Electronic identity documents (eIDs) are standard documents with an electronic layer capable of storing data and executing cryptographic protocols. The most widely used eID is the biometric passport (ePassport) introduced by the International Civil Aviation Organization (ICAO) and issued in more than 150 countries [180]. According to EU regulation, 2019/1157 [152], all European Union members must include an application supporting the ICAO in their national identity documents, making this the de facto standard for eIDs. Below we will give a high-level overview of the cryptographic protocols included in the ICAO standard.

Basic Access Control (BAC) is a password-based mechanism designed to thwart both online skimming and offline eavesdropping attacks. Attackers without the knowledge of the password (document number, date of birth, and expiry date) will be unable to read the passport's content and decipher the eavesdropped communication. However, the security of the BAC suffers from offline dictionary attacks due to the low entropy of the password. Its successor, Password Authentication Connection Establishment (PACE), provides a much better security guarantee by employing a Diffie-Hellman key.

Passive Authentication (PA) enables the reader to verify the authenticity of the eID data. During PA, the reader will retrieve, in addition to the data, the document security object (DSO), which encompasses a signature on the hash values of the data. The reader can then verify the authenticity of the data by comparing the hashed values

and verifying the signature using the issuer's public key. It is worth noting that the data is stored in so-called data groups (DG). This data contains personal data and random numbers (e.g., eID number). Moreover, the hash value of the data groups for the same personal data will be different, i.e., we assume that $\mathsf{H}(DG_1)$ and $\mathsf{H}(DG_2)$ are unlinkable, if only knowing the personal data and not the random numbers.

Chip Authentication (CA) prevents the cloning of eID and its data. Once read, the data could potentially be uploaded to a fresh eID, practically cloning the original eID. The problem is solved via a hardware assumption. We assume that the secret key loaded to the eID during the personalization phase cannot be extracted. The corresponding public key is added to the signed data, creating a link between it and the device. During verification, the reader checks that the public key used by the eID during this additional step is part of the data verified during passive authentication.

5.2.2 Anonymous Credentials

Anonymous credential systems (ACs) are a cryptographic building block envisioned and introduced by Chaum [47]. They allow users to obtain digital credentials encoding their attributes from an issuer. Users can later use those credentials to prove certain claims (e.g., over 18 years old) without revealing any other meaningful information about themselves.

Anonymous credentials have found applications in various problems and environments. Those include keyed-verification ACs [46; 55], AC as delegated parts of the credential to other parties [34; 36; 56], AC in the decentralized [80; 173] or cloud-based [118] setting. An interesting system, which can be treated as a single-use, single-attribute credential, is Privacy Pass [59]. It was introduced as a way to solve the CAPTCHA problem that anonymous network users are facing. There also exists a rate-limiting version [103] (introduced in iOS 16 [†]), which uses a trusted mediator to enforce the issuer's policy. The privacy guarantees are that the mediator does not learn the origin where the user will redeem the token. At the same time, the issuer

[†] https://developer.apple.com/videos/play/wwdc2022/10077

is oblivious to any data identifying the user, e.g., IP address or other distinguishing metadata. The exciting part of this chapter is that in this setting, secure hardware components of the iPhone, together with iCloud, play the role of the trusted mediator. We will later see that the architecture of our solution is similar to this.

Recently, Rosenberg et al. [167] introduced the idea of using the blockchain consensus to make anonymous credential issuers obsolete. The idea is to use an existing identity infrastructure and store the credentials in a secure data structure. They use a bulletin board based on Ethereum smart contracts for storage in their implementation. The exciting part related to our work is that they focus on using the existing infrastructure for eIDs, particularly the ICAO-based ePassport. They use the fact that the personal data stored on the ePassport is authenticated by a governmental authority via passive authentication. Unfortunately, their solutions fail to provide an active authentication of the ePassport. In particular, knowing the data stored on the eID and the DSO is enough to create the credentials in their system. However, this data is fully read during border control, making their system unusable.

Most notable in the context of our contribution is the very recent work by Schwarz et al. [170] in which the trusted execution environment (TEE) is used for the AC. The authors propose FeIDo, a TEE-based roaming FIDO authenticator that computes FIDO credentials based on user attributes. Their main goal is to solve the token loss problem since the same keys can be accessed using a different eID because it stores the same attributes. Interestingly, they notice that since the TEE gets access to the user's data, it can enforce access policies. Our approach is very different. Firstly, we support any FIDO token and are not bound to a custom instantiation of the token. Secondly, we are not bound to only electronic identity documents but also support standalone anonymous credential systems or any other, e.g., cloud-based identity systems with the non-shareability property. Lastly and most importantly, in our approach, the personal data never leaves the user's device, which is not true for [170].

5.3 Requirements and Threat Model

The main objective of FIDO-AC is to provide a practical system capable of augmenting the FIDO2 protocol with anonymous credentials derived from a verifiable source (i.e., eID). The system guarantees that the data was gathered from a legitimate document at the time of a FIDO transaction, and only selected information about the user is shared with relying party. The requirements are crafted with the criteria listed below:

R.1 Privacy Preserving. At the end of the FIDO-AC protocol, the relying party should learn only the relevant authenticated user information without compromising user privacy. In particular, FIDO2 privacy guarantees should not be violated.

R.2 Active Authentication (Liveliness). The FIDO-AC system should verify the possession of a non-sharable credential/device for the presented user attributes.

R.3 Compatibility. FIDO-AC should be fully compatible with the FIDO2 protocol.

R.4 User-Centric Design. The solution should impose minimal user friction to ease the adoption of FIDO-AC.

R.5 Pluggable Integration. The integration of the FIDO-AC system with an existing FIDO2 deployment should be effortless in terms of development, operation, and deployment. In particular, FIDO-AC should work without modification of existing FIDO clients and authenticators.

R.6 Efficient architecture. The FIDO-AC system should deliver reasonable performance (compared to the pure FIDO system), and it should be trivial to scale. Moreover, the architecture should be vendor agnostic.

We design FIDO-AC to be an extension of the existing FIDO2 standard with the extra capability of providing anonymous credentials. The threat model of FIDO2 web authentication [25] is used and extended to include new elements unique to the proposed system. Following the original FIDO trust assumptions, we assume authenticated communication channels between different parties. The integrity of the client agent, browser, authenticator, and OS is trusted, which is the same trust assumption needed for FIDO authentication. Additionally, to accommodate the introduction of

new modules, we trust the mediator party to perform the verification correctly and not collude with relying parties to link users. Moreover, we trust the eID infrastructure and the hardware-based protection of the eID device.

We assume the integrity of the underlying device hardware is correct and trusted. Side-channel attacks such as fault injection, power analysis, and micro-architecture side-channel attacks are also out-of-scope in this work. We also do not consider denial-of-service (DoS) attacks on the mediator. Last but not least, we assume the FIDO-AC mobile applications and services are free of software vulnerability.

5.4 Passwordless Authentication with Attributes

In this section, we introduce passwordless authentication with attributes (PAwA). The passwordless authentication protocol (PLA) captures the syntax of WebAuth and was formally introduced and analyzed in [32] and subsequently in [99]. However, the PLA protocol does not capture the notion of attributes, which is needed for FIDO-AC. Therefore, we extend the definitions from [99] to formally introduce attributes to the passwordless authentication.

In other words, we formally model how to use attributes in the FIDO framework. Unfortunately, it turns out that achieving this model with existing FIDO and credential systems while simultaneously fulfilling the requirements defined in the previous section is hard. Therefore, we will further extend this model by introducing an additional trusted party to the system called a mediator. This new party will act as a sort of interface between FIDO and the credential system. It will also introduce new privacy concerns, which will be addressed by formally introducing the mediator into our definitions of PAwA.

5.4.1 Formal Model of PAwA

The PAwA protocol consists of two processes, namely the registration phase and the authentication phase. The messages passed between a server and a token are relayed through an intermediary client interface, e.g., the web browser. In both phases, the

server sends the first message containing a challenge and the desired attribute policy. We follow the same syntax as the one used in [32; 99] and encapsulate the policy as part of the server's challenge [‡]. After receiving the challenge, the client, together with the token, computes the response, similar to standard PLA. Depending on the instantiation of PAwA, the client can either forward the attribute policy to the token or compute this part of the assertion locally.

In the registration phase, the token additionally attaches a public key pk and a credential identifier cid. The server keeps track of the information received from the token in its storage. Then, in subsequent authentication, the server includes the cid in the first message described above. In PAwA, the server verifies the signed message with respect to the augmented challenge. The server also verifies the response with respect to its attribute policy.

Formal Syntax

The model considers parties $\mathcal{P} = \mathcal{T} \cup \mathcal{S}$, where the parties are partitioned into the set of tokens $\mathcal{T}$ and the set of servers $\mathcal{S}$. Each token $T \in \mathcal{T}$ has a fixed state that is initialized with a key msk_T. Additionally, we associate an attribute set Att_T with each token T, where attributes are elements from a set Att_U. Each server $S \in \mathcal{S}$ constructs a key-value table known as the registration context rcs_S, whereby a new entry will be inserted whenever a token registers with the server. Each server is also uniquely identified with its publicly known unique identifier $id_\mathcal{S}$, which in practice, corresponds to a URL. The server is also assumed to know the user account and its cid. Moreover, each server specifies an access policy $\mathsf{Policy}_S \subseteq \mathsf{Att}_\mathsf{U}$.

The syntax of the PAwA protocol is formally defined in Definition 1. To model the capability of the adversary to freely communicate with tokens and servers, a server oracle and a token oracle are additionally defined in Definition 3, whereby st_S is used to model the state transfer between algorithms, C_s is used to bind registration to authentication, and the $\pi^{i,j}$ handle is used to model the instances of registration and

[‡]Note that this is in line with how one would implement the attribute policies as part of the extension fields of a FIDO challenge.

authentication. Partnering of two handles $\pi_S^{i,j}$ and $\pi_T^{i',j'}$ as defined in Definition 4, for which they share the same session identifier, is used as the winning condition of the security experiments.

Definition 1 (PAwA) *A passwordless authentication scheme with attributes (*PAwA*) is a tuple* $\mathsf{PAwA} = (\mathsf{Gen}, \mathsf{Reg}, \mathsf{Auth})$*:*

- Gen*: on input parameters* par*, outputs a secret key* msk*.*
- Reg*: given as a tuple of the following algorithms:*

  $\mathsf{rchall}_{\mathsf{ac}}$**:** *on input of a server identity* id_S*, outputs challenge with policy value* c_p *and a state* st*.*

  $\mathsf{rcomm}_{\mathsf{ac}}$**:** *on input of a server identity* id_S *and a challenge* c_p*, outputs a message* M_r*.*

  $\mathsf{rresp}_{\mathsf{ac}}$**:** *on input of a master secret key* msk*, a server identity* id_S *and a message* M_r*, outputs credential identifier* cid *and a response* R_r*.*

  $\mathsf{rcheck}_{\mathsf{ac}}$**:** *on input of a state* st*, a credential identifier* cid *and a response* R_{ac}[§]*, outputs a bit* b *and a credential* cred*.*
- Auth*: given as a tuple of the following algorithms:*

  $\mathsf{achall}_{\mathsf{ac}}$**:** *on input of a server identity* id_S*, outputs a challenge* c_p *and a state* st*.*

  $\mathsf{acomm}_{\mathsf{ac}}$**:** *on input of a server identity* id_S *and a challenge* c_p*, outputs a message* M_a*.*

  $\mathsf{aresp}_{\mathsf{ac}}$**:** *on input of a master secret key* msk*, a server identity* id_S*, a credential identifier* cid*, and a message* M_a*, outputs a response* R_a*.*

  $\mathsf{acheck}_{\mathsf{ac}}$**:** *on input of a state* st*, a registration context* rcs*, a credential identifier* cid *and a response* R_{ac}[§]*, outputs a bit* $b \in \{0, 1\}$

Algorithms $\mathsf{rchall}_{\mathsf{ac}}$ *,* $\mathsf{rcheck}_{\mathsf{ac}}$*,* $\mathsf{achall}_{\mathsf{ac}}$*,* $\mathsf{acheck}_{\mathsf{ac}}$ *are executed by servers,* $\mathsf{rcomm}_{\mathsf{ac}}$*,* $\mathsf{acomm}_{\mathsf{ac}}$ *are executed by clients, and* $\mathsf{rresp}_{\mathsf{ac}}$*,* $\mathsf{aresp}_{\mathsf{ac}}$ *are executed by tokens.*

Definition 2 (Policy Extraction) *We assume that there exists a function* $\mathsf{Pol} - \mathsf{Ext}(M)$ *that on input of the message outputs of* $\mathsf{acomm}_{\mathsf{ac}}(\mathsf{id}_S, \cdot)$*, and* $\mathsf{rcomm}_{\mathsf{ac}}(\mathsf{id}_S, \cdot)$ *returns the policy* Policy_S*.*

[§]Depending on the flow type, authentication or registration, R_{ac} contains either R_a or R_r

Definition 3 (Server and Token Oracles) *Let $\mathcal{A}$ be an adversary algorithm and* $\mathsf{PAwA} = (\mathsf{Gen}, \mathsf{Reg}, \mathsf{Auth})$ *be a passwordless authentication with attributes scheme. Each party $P \in \mathcal{T} \cup \mathcal{S}$ is associated with a set of handles $\pi_P^{i,j}$ that models two types of instances corresponding to registration and authentication. Each party is represented by a number of these instances. Concretely, $\pi_P^{i,j}$ for $j = 0$ is the i-th registration instance of party P and for $j \geqslant 1$ is the j-th authentication instance of P corresponding to the i-th registration.*

It is assumed that for each token $T \in \mathcal{T}$, a secret key is generated as $\mathsf{msk}_T \leftarrow \mathsf{Gen}(\mathsf{par})$. *For each server $S \in \mathcal{S}$, key-value tables* $\mathsf{rcs}_S, C_S, \mathsf{st}_S$ *are given. By default, these are empty. Adversary, $\mathcal{A}$ has access to oracles* $\mathsf{Setup}, \mathsf{Start}, \mathsf{Challenge}, \mathsf{Complete}$ *defined as follows:*

- $\mathsf{Setup}(\mathsf{Policy}_{LS}, \mathsf{Att}_{LT})$*: Executes* $(\mathsf{Policy}_{S_1}, ..., \mathsf{Policy}_{S_n}) := \mathsf{Policy}_{LS}$, *and* $(\mathsf{Att}_{T_1}, ..., \mathsf{Att}_{T_m}) := \mathsf{Att}_{LT}$.
- $\mathsf{Start}(\pi_S^{i,j})$*: This executes* $(c_p, \mathsf{st}) \leftarrow \mathsf{rchall}_{\mathsf{ac}}(\mathsf{id}_S)$ *in case* $j = 0$ *or* $(c_p, \mathsf{st}) \leftarrow \mathsf{achall}_{\mathsf{ac}}(\mathsf{id}_S)$ *in case* $j > 0$. *The oracle sets* $\mathsf{st}_S[i,j] := \mathsf{st}$ *and returns c_p to $\mathcal{A}$.*
- $\mathsf{Challenge}(\pi_T^{i,j}, \mathsf{id}_S, \mathsf{cid}, M)$*: Executes* $(\mathsf{cid}, R_r) \leftarrow \mathsf{rresp}_{\mathsf{ac}}(\mathsf{msk}_T, \mathsf{id}_S, M)$ *if* $j = 0$ *or* $R_a \leftarrow \mathsf{aresp}_{\mathsf{ac}}(\mathsf{msk}_T, \mathsf{id}_S, \mathsf{cid}, M)$ *if* $j > 0$. *$\mathcal{A}$ is given (*(cid, R_r) *or R_a) .*
- $\mathsf{Complete}(\pi_S^{i,j}, \mathsf{cid}, R)$*: Aborts if* $\mathsf{Start}(\pi_S^{i,j})$ *has not been queried before. If* $j = 0$, *it executes* $(b, \mathsf{cred}) \leftarrow \mathsf{rcheck}_{\mathsf{ac}}(\mathsf{st}_S[i,j], \mathsf{cid}, R)$, *sets* $C_S[i] := \mathsf{cid}$, *and* $\mathsf{rcs}_S[\mathsf{cid}] := \mathsf{cred}$. *If* $j > 0$, *it aborts if* $\mathsf{cid} \neq C_S[i]$. *Otherwise, it executes* $b \leftarrow \mathsf{acheck}_{\mathsf{ac}}(\mathsf{st}_S[i,j], \mathsf{rcs}_S, \mathsf{cid}, R)$. *In both cases, b is returned to $\mathcal{A}$.*

It is assumed that for each $(i, j, T, S) \in \mathbb{N} \times \mathbb{N} \times \mathcal{T} \times \mathcal{S}$, *the oracles* $\mathsf{Setup}(\cdot,\cdot)$, $\mathsf{Start}(\pi_S^{i,j})$, $\mathsf{Challenge}(\pi_T^{i,j}, \cdot, \cdot, \cdot)$, *and* $\mathsf{Complete}(\pi_S^{i,j}, \cdot, \cdot)$ *are executed only once.*

Definition 4 (Session Identifiers and Partnering) *Consider the oracles from Definition 3. Let V_t be a function that takes as input the transcript* $tr_T^{i,j} = (\mathsf{id}_S, \mathsf{cid}, M, R)$ *that a token $T \in \mathcal{T}$ observes in an oracle call to* $\mathsf{Challenge}(\pi_T^{i,j}, \cdot, \cdot, \cdot)$, *and outputs a bitstring $V_t(tr_T^{i,j})$. Similarly, let V_s be a function that takes as input the transcript* $tr_S^{i,j} = (c, \mathsf{cid}, R)$ *that a server $S \in \mathcal{S}$ observes in oracle calls to* $\mathsf{Start}(\pi_S^{i,j})$, $\mathsf{Complete}(\pi_S^{i,j}, \cdot, \cdot)$, *and outputs a bitstring $V_s(tr_S^{i,j})$. It is assumed that these functions*

are specified by PAwA. *The handles* $\pi_T^{i,j}$ *and* $\pi_S^{i',j'}$ *are partnered if:* $(j = 0 \iff j' = 0) \wedge V_t(tr_T^{i,j}) = V_s(tr_S^{i',j'})$.

Security and Privacy

We will now define what it means for a PAwA protocol to be secure. We begin with security against impersonation, which informally ensures that there is precisely one partnered session for an accepting server. The security is defined in Definition 5, for which the adversary can interact with tokens and servers concurrently by using the oracles defined in Definition 3.

Definition 5 (Impersonation Security (Adapted from [99])) *For a* PAwA = (Gen, Reg, Auth) *scheme, the following security experiment* $\mathbf{Imp}^{\mathcal{A}}_{\mathsf{PAwA}}$ *is defined to run between the challenger and an adversary* $\mathcal{A}$.

- ***Setup.*** *For each token* $T \in \mathcal{T}$, *a key is generated by running* $\mathsf{msk}_T \leftarrow \mathsf{Gen}(\mathsf{par})$. *The adversary assigns the attribute set for tokens and policy set for servers by calling the oracle* Setup *and passing in the attribute set lists.*
- ***Online Phase.*** *The adversary is allowed to interact with the oracles* Start, Challenge, Complete *as in Definition 3.*
- ***Output Phase.*** *Finally,* $\mathcal{A}$ *terminates, and the experiment outputs* 1 *if and only if there exists a server handle* $\pi_S^{i,j}$ *for* $j > 0$ *such that the following conditions hold:*
  1. $\pi_S^{i,0}$ *is partnered with a token handle* $\pi_T^{k,0}$.
  2. $\pi_S^{i,j}$ *accepted, i.e. in call* $\mathsf{Complete}(\pi_S^{i,j}, \mathsf{cid}, R)$, *algorithm* $\mathsf{acheck}(\mathsf{st}_S[i,j], \mathsf{rcs}_S, \mathsf{cid}, R)$ *returned* 1.
  3. $\pi_S^{i,j}$ *is not partnered with any token handle* $\pi_T^{i',j'}$, *or it is partnered with a token handle, which is partnered with a different server handle* $\pi_{S'}^{i'',j''}$.

In addition to the standard FIDO security defined above, we will now define attribute unforgeability. Informally, we want to ensure that an adversary can only access the server if it possesses a token that adheres to the server's policy. We achieve this by

requiring that if S accepts, then the partnered token must have the attribute set that satisfies the server policy.

Definition 6 (Attribute Unforgeability) *For a* $\mathsf{PAwA} = (\mathsf{Gen}, \mathsf{Reg}, \mathsf{Auth})$ *scheme, the following security experiment* $\mathbf{Att\text{-}Unf}^{\mathcal{A}}_{\mathsf{PAwA}}$ *is defined to run between the challenger and an adversary* $\mathcal{A}$*.*

- ***Setup.*** *For each token* $T \in \mathcal{T}$*, a key is generated by running* $\mathsf{msk}_T \leftarrow \mathsf{Gen}(\mathsf{par})$*. The adversary assigns the attribute set for tokens and policy set for servers by calling the oracle* Setup *and passing in the attribute set lists.*
- ***Online Phase.*** *The adversary is allowed to interact with the oracles* Start, Challenge, Complete *as in Definition 3.*
- ***Output Phase.*** *Finally,* $\mathcal{A}$ *terminates, and the experiment outputs* 1 *if and only if there exists a server handle* $\pi_S^{i,j}$ *for* $j > 0$ *such that the following conditions hold:*

  1. $\pi_S^{i,0}$ *is partnered with a token handle* $\pi_T^{k,0}$*.*
  2. $\pi_S^{i,j}$ *accepted, i.e., in call* $\mathsf{Complete}(\pi_S^{i,j}, \mathsf{cid}, R)$*, algorithm* $\mathsf{acheck}(\mathsf{st}_S[i,j], \mathsf{rcs}_S, \mathsf{cid}, R)$ *returned* 1*.*
  3. $\pi_S^{i,j}$ *is not partnered with any token handle* $\pi_T^{i',j'}$*, or it is partnered with a token handle for which its attribute set* Att_T *does not satisfy the server policy* Policy_S*.*

Similarly to the standard FIDO security model, we introduce an unlinkability definition that ensures that tokens are not linkable across origins. We extend the unlinkability proposed in [99] to capture attributes. Informally, we ensure that the server cannot learn more attributes than its policy has requested. In the unlinkability experiment, the adversary $\mathcal{A}$ is given access to the oracles defined in Definition 7, and with it, $\mathcal{A}$ gains the global view of the system. $\mathcal{A}$ is given two additional oracles Left and Right, which run T_b and T_{1-b} for a random bit b. $\mathcal{A}$ is said to win the game if it can determine which token is used in which oracle with respect to the conditions of instance freshness and credential separation defined in Definition 3. Credential separation models attack

that a server can launch when the same token is used twice at the same server, while instance freshness is a consequence of the oracles definition.

Definition 7 (Unlinkability (Adapted from [99])) *For a* $\mathsf{PAwA} = (\mathsf{Gen}, \mathsf{Reg}, \mathsf{Auth})$*, following experiment* $\mathbf{Unl}^{\mathcal{A}}_{\mathsf{PAwA}}$ *is defined to run between the challenger and an adversary* $\mathcal{A}$*.*

- ***Setup.*** *For each token* $T \in \mathcal{T}$*, a key is generated by running* $\mathsf{msk}_T \leftarrow \mathsf{Gen}(\mathsf{par})$*. The adversary assigns the attribute set for tokens and policy set for servers by calling the oracle* Setup *and passing in the attribute set lists.*
- ***Phase 1.*** *The adversary is allowed to interact with oracles* Start, Challenge, Complete *(see Definition 3). Moreover, we allow the adversary to query the* Challenge *oracle in a way that the oracle also executes the client part of the execution, i.e., the* Challenge *oracle additionally executes the* rcomm_{ac} *or* acomm_{ac} *algorithms.*
- ***Phase 2.*** *The adversary outputs two (not necessarily distinct) token identifiers* T_0, T_1*, and two (not necessarily distinct) server identifiers* $S_L, S_R \in \mathcal{S}$ *such that:*

$$\mathsf{Att}_{T_0} \supseteq \mathsf{Policy}_S \iff \mathsf{Att}_{T_1} \supseteq \mathsf{Policy}_S,$$

for all $S \in \{S_L, S_R\}$*. Let* i_0 *and* i_1 *be the smallest identifiers for which the token handles* $\pi^{i_0,0}_{T_0}$ *and* $\pi^{i_1,0}_{T_1}$ *were not queried to the* Challenge *oracle in Phase 1. The experiment chooses a bit* b *uniformly at random. It sets* $j_0 := 0, j_1 := 0$ *and initializes two oracles* Left, Right *as follows:*

  - $\mathsf{Left}(\mathsf{cid}, M)$*: Abort if* $\mathsf{Pol} - \mathsf{Ext}(M) \neq \mathsf{Policy}_{S_L}$*, else return* $\mathsf{Challenge}(\pi^{i_b,j_b}_{T_b}, \mathsf{id}_{S_L}, \mathsf{cid}, M)$ *and set* $j_b = j_b + 1$*.*
  - $\mathsf{Right}(\mathsf{cid}, M)$*: Abort if* $\mathsf{Pol} - \mathsf{Ext}(M) \neq \mathsf{Policy}_{S_R}$*, else return* $\mathsf{Challenge}(\pi^{i_{1-b},j_{1-b}}_{T_{1-b}}, \mathsf{id}_{S_R}, \mathsf{cid}, M)$ *and set* $j_{1-b} = j_{1-b} + 1$*.*

  Like in Phase 1, we allow the adversary to decide if the Left *and* Right *oracles should execute the client part algorithms.*
- ***Phase 3.*** *The adversary can interact with all the oracles defined in Phases 1*

and 2.

- ***Output Phase.*** *Finally, the adversary outputs a bit* $\hat{b}$*. Consider the following lists of* cid*'s:*

  - $\mathcal{L}^r_{ch}$ *contains all* cid*'s returned by queries that are not issued via* Left, Right *and are of the form* $\mathsf{Challenge}(\pi^{i,0}_T, \mathsf{id}_S, \cdot, \cdot)$ *for any* $i, T \in \{T_0, T_1\}$ *and* $S \in \{S_L, S_R\}$.

  - $\mathcal{L}^a_{ch}$ *contains all* cid*'s that are part of the input of queries that are not issued via* Left, Right *and are of the form* $\mathsf{Challenge}(\pi^{i,j}_T, \mathsf{id}_S, \cdot, \cdot)$ *for any* $j > 0$,$i, T \in \{T_0, T_1\}$ *and* $S \in \{S_L, S_R\}$.

  - $\mathcal{L}^r_{lr}$ *contains all* cid*'s returned by queries to* Left *or* Right *when* $j_b = 0$ *or* $j_{1-b} = 0$*, respectively.*

  - $\mathcal{L}^a_{lr}$ *contains all* cid*'s that are part of the queries to* Left *or* Right *when* $j_b > 0$ *or* $j_{1-b} > 0$*, respectively.*

  The experiment returns 1 *if and only if:*
  - *bit* $\hat{b}$ *is equal to bit* b*, and*
  - *(instance freshness) the adversary never made a query to oracle* Challenge *using handles* $\pi^{i_0,k_0}_{T_0}$ *and* $\pi^{i_1,k_1}_{T_1}$ *for any* k_0, k_1*, and*
  - *(credential separation) The following set is empty:*

$$\begin{aligned}
\mathbf{wUnl}: &\ (\mathcal{L}^r_{ch} \cup \mathcal{L}^a_{ch}) \cap (\mathcal{L}^r_{lr} \cup \mathcal{L}^a_{lr}) \\
\mathbf{mUnl}: &\ ((\mathcal{L}^r_{ch} \cup \mathcal{L}^a_{ch}) \cap \mathcal{L}^a_{lr}) \cup ((\mathcal{L}^r_{lr} \cup \mathcal{L}^a_{lr}) \cap \mathcal{L}^a_{ch}) \\
\mathbf{sUnl}: &\ (\mathcal{L}^r_{ch} \cap \mathcal{L}^a_{lr}) \cup (\mathcal{L}^r_{lr} \cap \mathcal{L}^a_{ch})\,.
\end{aligned}$$

Depending on credential separation, we distinguish three levels of unlinkability: weak, medium and strong.

5.4.2 Formal Model of PAwAM

In the previous section, we introduced a security model for FIDO with attributes. Unfortunately, introducing attributes without significant changes to the FIDO specification and token firmware is impossible, which violates our compatibility requirement **R.3** and pluggable integration requirement **R.5**.

Our goal is to build a system that uses existing building blocks. In particular, we want to interface existing FIDO solutions with attribute-based systems (e.g., anonymous credentials or ICAO eID-based attributes). To this end, we must introduce a trusted party called a mediator that acts as the interface between both systems.

Formal Syntax

The mediator is identifiable using a public key pk_M with a corresponding secret key sk_M. This new party introduces additional security problems, which we capture formally in the definitions below. In our syntax, we will also use $\mathsf{ask}_T \leftarrow \mathsf{IssCred}(\mathsf{Att}_T)$ to denote a token-specific secret key for the attribute-based credential systems. To simplify our considerations, we abstract the attribute-based system issuing process using algorithm $\mathsf{IssCred}$. We assume this algorithm outputs a fresh ask for the given attributes, leading to fresh credentials for the attribute-based system (e.g., new data groups in case of eID systems). It is worth noting that in PAwA implementations, the ask_T can be part of the token's master secret key. We make this key explicit in PAwAM to simplify the description. The user platform (i.e., token and client) can use this key to prove possession of attributes to the mediator. We assume that this key implicitly defines the attributes Att_T corresponding to token T.

Definition 8 (PAwAM) *A passwordless authentication scheme with attributes and mediators (*PAwAM*) is a tuple* $\mathsf{PAwAM} = (\mathsf{Gen}, \mathsf{Reg}, \mathsf{Auth}, \mathsf{Med})$*:*

- $\mathsf{Gen}, \mathsf{Reg}, \mathsf{Auth}$*: has the same description as in* PAwA*.*
- Med*: given as a tuple of the following algorithms:*

$\mathsf{attestreq}_{\mathsf{ac}}$: *on input of a secret key* ask_T*, a server challenge* c*, outputs an attestation request* req_M *and* nonce.

$\mathsf{attestchal}_{\mathsf{ac}}$: *on input of a request* req_M*, a mediator secret key* sk_M*, a mediator public key* pk_M*, outputs an attestation state* $\mathsf{st}_{\mathsf{chal}}$ *and a challenge* chal_M.

$\mathsf{attestresp}_{\mathsf{ac}}$: *on input of an attribute secret key* ask_T*, a challenge* chal_M*, outputs an attestation response* resp_M.

$\mathsf{attest}_{\mathsf{ac}}$: *on input of a challenge state* $\mathsf{st}_{\mathsf{chal}}$*, a response* resp_M *and a mediator secret key* sk_M*, outputs a attestation message* att_m *and a signature* σ_m.

$\mathsf{prove}_{\mathsf{ac}}$: *on input of an attestation message* att_m*, an attestation signature* σ_m*, a nonce* nonce*, attributes* Att*, a policy* Policy_S*, outputs a proof of attribute possession* Π_{Att}.

$\mathsf{check}_{\mathsf{ac}}$: *on input of a proof* Π_{Att}*, a policy* Policy_S*, a mediator public key* pk_M*, and a challenge* c*, outputs a bit* b_{ac}.

Algorithms $\mathsf{check}_{\mathsf{ac}}$ *is executed by servers during the execution of* $\mathsf{rcheck}_{\mathsf{ac}}$ *and* $\mathsf{acheck}_{\mathsf{ac}}$. *Algorithms* $\mathsf{attestchal}_{\mathsf{ac}}$ *and* $\mathsf{attest}_{\mathsf{ac}}$ *are executed by mediators,* $\mathsf{attestreq}_{\mathsf{ac}}$, $\mathsf{attestresp}_{\mathsf{ac}}$ *and* $\mathsf{prove}_{\mathsf{ac}}$ *are executed by clients.*

Definition 9 (Oracles) *For* PAwAM*, we use the same server and token oracle defined for* PAwA*. We slightly modify the* Setup *oracle, which now additionally sets the keys* ask_T *of tokens according to the attributes the oracle sets. Additionally, we allow the adversary to communicate with the mediator* M^i*, where* i *represents the* i*-th session of the mediator using the following oracles:*

- $\mathsf{MedReq}(T, c)$*: The oracle executes* $(\mathsf{nonce}, \mathsf{req}_M) \leftarrow \mathsf{attestreq}_{\mathsf{ac}}(\mathsf{ask}_T, c)$*. The result is returned to* $\mathcal{A}$.
- $\mathsf{MedChal}(M^i, \mathsf{req}_M)$*: It executes* $\mathsf{st}_{\mathsf{chal}}, \mathsf{chal}_M \leftarrow \mathsf{attestchal}_{\mathsf{ac}}(\mathsf{req}_M, \mathsf{sk}_M, \mathsf{pk}_M)$ *and returns* chal_M *to* $\mathcal{A}$ *and sets* $st_M[M^i] := \mathsf{st}_{\mathsf{chal}}$.
- $\mathsf{MedResp}(T, \mathsf{chal}_M)$*: The oracle executes* $\mathsf{resp}_M \leftarrow \mathsf{attestresp}_{\mathsf{ac}}(\mathsf{chal}_M, \mathsf{ask}_T)$ *and forwards the output to* $\mathcal{A}$.
- $\mathsf{MedAttest}(M^i, \mathsf{resp}_M)$*: The oracle executes* $\mathsf{st}_{\mathsf{chal}} := st_M[M^i]$ *and* $(att_m, \sigma_m) \leftarrow \mathsf{attest}_{\mathsf{ac}}(\mathsf{st}_{\mathsf{chal}}, \mathsf{resp}_M, \mathsf{sk}_M)$*. The result* (att_m, σ_m) *is returned to* $\mathcal{A}$.

Security and Privacy of PAwAM

We will now define the security experiment for passwordless authentication with attributes and a mediator. All definitions follow the same pattern as in the standard PAwA case, except that we provide the adversary with means to simulate the interaction between tokens and the mediator. As mentioned, we must also introduce security notions that will capture a malicious mediator that tries to break the system's privacy (e.g., learning the attributes or the origin of the server).

Definition 10 (Impersonation Security) *For the* PAwAM*, the impersonation security experiment,* $\mathbf{Imp}^{\mathcal{A}}_{\mathsf{PAwAM}}$ *is the same as defined for* PAwA*, except that the adversary* $\mathcal{A}$ *is given access to the oracles from Definition 9 during the online phase.*

Definition 11 (Attribute Unforgeability) *For* PAwAM*, the attribute unforgeability experiment,* $\mathbf{Att\text{-}Unf}^{\mathcal{A}}_{\mathsf{PAwAM}}$ *is the same as defined for* PAwA*, except that* $\mathcal{A}$ *is given access to the oracles from Definition 9 during the online phase. We define as an additional winning condition that there is no query made to* MedReq *using challenge* c *for the server handle* $\pi_S^{i,j}$*.*

Definition 12 (Unlinkability) *For the* PAwAM*, the unlinkability experiment,* $\mathbf{Unl}^{\mathcal{A}}_{\mathsf{PAwAM}}$ *is the same as defined for* PAwA*, except that the adversary* $\mathcal{A}$ *is given access to the oracles from Definition 9 during the phases 1 and 3.*

We will now introduce two notions for PAwAM that informally capture the following two privacy concerns. First, we ensure that given an attestation request, the mediator cannot distinguish which origin the user is trying to access. Second, the mediator should attest to the user's attributes without learning anything about them. We capture the first notion informally with an experiment where the adversary specifies one token and two servers. The experiment then proceeds with picking one of the servers and starting the interaction with the adversary that plays the role of the mediator. The adversary wins if it can guess which server the user tried to access. We call this notion origin privacy and define it more formally in Definition 13. In addition to origin privacy, we define attribute privacy that captures the latter informal property. The adversary

now picks two tokens and one server. We want to model that no information about the attributes of the two tokens is leaked to the mediator. Therefore, the experiment randomly picks one of the tokens and asks the adversary to attest the token to the chosen server. Here we distinguish two versions of attribute privacy: one-time and many-time. Both ensure that the attributes used are hidden from the mediator, but in one-time privacy, attestation requests from the same token are linkable. Notably, this is similar to one-time-show and multi-show security of anonymous credentials, where in the former showing attributes twice is linkable. Still, the attributes (e.g., personal data) are hidden, and there is no link to the issuing process of the credentials. Although the mediator can see that the same token/user is trying to receive an attestation, the server cannot do this. Moreover, implementing a local mediator is a simple solution to make any one-time scheme many-time secure. In Section 5.6.8, we provide threat analysis that depends on how the mediator is implemented in practice.

We assume that the mediator does not collude with the servers or the tokens, and further security analysis of the collusion is provided in Section 5.6.8.

Definition 13 (Origin Privacy) *For the* PAwAM*, the origin privacy experiment,* $\textbf{Orig-Priv}^{\mathcal{A}}_{\mathsf{PAwAM}}$ *is defined to run between the challenger and an adversary* $\mathcal{A}$*.*

- ***Setup.*** *For each token* $T \in \mathcal{T}$*, a key is generated by running* $\mathsf{msk}_T \leftarrow \mathsf{Gen}(\mathsf{par})$*. The adversary assigns the attribute set for tokens and policy set for servers by calling the oracle* Setup *and passing in the attribute set lists.*
- ***Phase 1.*** *The adversary is allowed to interact with oracles* Start, Complete, Challenge *(see Definition 3) and oracles* MedReq, MedResp*.*
- ***Challenge Phase.*** *The adversary outputs a token identifier* T *and two (not necessarily distinct) server identifiers* $S_0, S_1 \in \mathcal{S}$*. The experiment chooses a bit* b *uniformly at random and runs* $(c_b, \cdot) \leftarrow \mathsf{achall}_{\mathsf{ac}}(\mathsf{ask}_T, \mathsf{id}_{S_b})$ *and* $(\mathsf{nonce}_b, \mathsf{req}_{M,b}) \leftarrow \mathsf{attestreq}_{\mathsf{ac}}(\mathsf{ask}_T, c_b)$*. The experiment gives* $\mathsf{req}_{M,b}$ *to the adversary. The adversary outputs a challenge* $\mathsf{chal}_{M,b}$ *to which the experiment responds with* $\mathsf{resp}_{M,b}$*, where* $\mathsf{resp}_{M,b} \leftarrow \mathsf{attestresp}_{\mathsf{ac}}(\mathsf{ask}_T, \mathsf{chal}_{M,b})$*.*
- ***Output Phase*** *Finally, the adversary outputs a bit* $\hat{b}$*. The experiment returns*

1 *if and only if bit* $\hat{b}$ *is equal to bit* b.

Definition 14 (One-time Attribute Privacy) *For the* PAwAM*, the attribute privacy experiment,* $\textbf{Att-Priv}^{\mathcal{A}}_{\mathsf{PAwAM}}$ *is defined to run between the challenger and an adversary* $\mathcal{A}$.

- ***Setup.*** *For each token* $T \in \mathcal{T}$*, a key is generated by running* $\mathsf{msk}_T \leftarrow \mathsf{Gen}(\mathsf{par})$. *The adversary assigns the attribute set for tokens and policy set for servers by calling the oracle* Setup *and passing in the attribute set lists.*
- ***Phase 1.*** *The adversary can interact with oracles* Start, Complete, Challenge *(see Definition 3). Additionally, it can interact with oracles* MedReq, MedResp.
- ***Challenge Phase 1.*** *The adversary outputs two (not necessarily distinct) token identifiers* T_0, T_1*, and one server identifier* $S \in \mathcal{S}$.
- ***Challenge Phase 2.*** *The experiment refreshes the attribute-based secret keys* $\mathsf{ask}_{T_0}, \mathsf{ask}_{T_1}$ *for tokens* T_0, T_1 *by running* $\mathsf{ask}_{T_0} \leftarrow \mathsf{IssCred}(\mathsf{Att}_{T_0})$ *and* $\mathsf{ask}_{T_1} \leftarrow \mathsf{IssCred}(\mathsf{Att}_{T_1})$.
- ***Challenge Phase 3.*** *The experiment chooses a bit* b *uniformly at random and runs:* $(c, \cdot) \leftarrow \mathsf{achall}_{\mathsf{ac}}(\mathsf{id}_S)$ *and* $(\mathsf{nonce}_b, \mathsf{req}_{M,b}) \leftarrow \mathsf{attestreq}_{\mathsf{ac}}(\mathsf{ask}_{T_b}, c)$. *The experiment gives* $\mathsf{req}_{M,b}$ *to the adversary. The adversary outputs a challenge* $\mathsf{chal}_{M,b}$ *to which the experiment responds with* $\mathsf{resp}_{M,b}$*, where* $\mathsf{resp}_{M,b} \leftarrow \mathsf{attestresp}_{\mathsf{ac}}(\mathsf{chal}_{M,b}, \mathsf{ask}_{T_b})$.
- ***Output Phase*** *Finally, the adversary outputs a bit* $\hat{b}$*. The experiment returns* 1 *if and only if bit* $\hat{b}$ *is equal to bit* b.

Definition 15 (Many-times Attribute Privacy) *We define many-times attribute privacy similar to one-time attribute privacy, except that it omits* ***Challenge Phase 2.***

5.5 FIDO-AC: System Design

In this section, we will describe the FIDO-AC system. First, we give an overview of actors and interactions. We then describe the requirements that an anonymous

credential system must support to be used in FIDO-AC. We show this in the example of an electronic identity document (eID) in the ICAO standard. Finally, we show how those credentials can be integrated into the FIDO2 authentication process.

$\underline{\textbf{attestreq}_{\textbf{ac}}(\mathsf{ask}_T, c)}$ $\mathsf{nonce} \xleftarrow{\$} \{0,1\}^\lambda$ $(\mathsf{H}(DG), pk_{eID}, \pi_{PA}) \leftarrow BAC/PACE(\mathsf{ask}_T)$ $\mathsf{req}_M := (\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}, c, \mathsf{nonce})$ $ret\ \mathsf{nonce}, \mathsf{req}_M$	$\underline{\textbf{attestchal}_{\textbf{ac}}(\mathsf{req}_M, \mathsf{sk}_M, \mathsf{pk}_M)}$ $(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}, c, \mathsf{nonce}) := \mathsf{req}_M$ $key_{ses} \leftarrow KE(\mathsf{pk}_{eID}, \mathsf{sk}_M)$ $cmd_{cha} \leftarrow AE\text{-}ENC(key_{ses}, cmd)$ $\mathsf{chal}_M := (\mathsf{pk}_M, cmd_{cha})$ $\mathsf{st}_{\mathsf{chal}} := (\mathsf{req}_M, key_{ses})$ $ret\ \mathsf{st}_{\mathsf{chal}}, \mathsf{chal}_M$
$\underline{\textbf{attestresp}_{\textbf{ac}}(\mathsf{chal}_M, \mathsf{ask}_T)}$ $(\mathsf{pk}_M, cmd_{cha}) := \mathsf{chal}_M$ $\mathsf{resp}_M \leftarrow CA(\mathsf{pk}_M, cmd_{cha}, \mathsf{ask}_T)$ $ret\ \mathsf{resp}_M$	$\underline{\textbf{attest}_{\textbf{ac}}(\mathsf{st}_{\mathsf{chal}}, \mathsf{resp}_M, \mathsf{sk}_M)}$ $(\mathsf{req}_M, key_{ses}) := \mathsf{st}_{\mathsf{chal}}$ $(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}, c, \mathsf{nonce}) := \mathsf{req}_M$ $b_{PA} \leftarrow PA_{verify}(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA})$ $b_{CA} \leftarrow CA_{Verify}(\mathsf{resp}_M, key_{ses})$ $att_m := \mathsf{H}(\mathsf{H}(DG)\|\mathsf{nonce})\|c$ $\sigma_m := \perp$ $\sigma_m \leftarrow \mathsf{Sign}(\mathsf{sk}_M, att_m)\ if\ (b_{PA} \wedge b_{CA})$ $ret\ att_m, \sigma_m$
$\underline{\textbf{prove}_{\textbf{ac}}(att_m, \sigma_m, \mathsf{nonce}, \mathsf{Att}, \mathsf{Policy}_S)}$ $DG \leftarrow Parse(\mathsf{Att})$ $(m\|c_m) := att_m$ $\pi_{\mathsf{zkp}} \leftarrow \mathsf{ZKProve}(\mathsf{crs}, (m, \mathsf{Policy}_S), (DG, \mathsf{nonce}))$ $\Pi_{\mathsf{Att}} := (att_m, \sigma_m, \pi_{\mathsf{zkp}})$ $ret\ \Pi_{\mathsf{Att}}$	$\underline{\textbf{check}_{\textbf{ac}}(\Pi_{\mathsf{Att}}, \mathsf{Policy}_S, \mathsf{pk}_M, c)}$ $(att_m, \sigma_m, \pi_{\mathsf{zkp}}) := \Pi_{\mathsf{Att}}$ $b_M \leftarrow \mathsf{Verify}(\mathsf{pk}_M, att_m, \sigma_m)$ $(m\|c_m) := att_m$ $b_{zkp} \leftarrow \mathsf{ZKVer}(\mathsf{crs}, (m, \mathsf{Policy}_S))$ $b_{challenge} \leftarrow c_m =^? c$ $b_{ac} \leftarrow b_M \wedge b_{zkp} \wedge b_{challenge}$ $ret\ b_{ac}$

TABLE 5.1: The pseudocode of FIDO-AC for the algorithms defined in Definition 8. Wrapper algorithms: KE - key exchange, $AE-ENC$ - authenticated encryption, PA_{verify} - passive authentication verification, CA_{verify} - chip authentication verification.

5.5.1 Overview

FIDO-AC is a novel system that satisfies the requirements and threat model stated in Section 5.3. Figure 5.2 illustrates the high-level overview of the new modules introduced by the FIDO-AC extension to the existing FIDO2. At its core, the FIDO-AC system

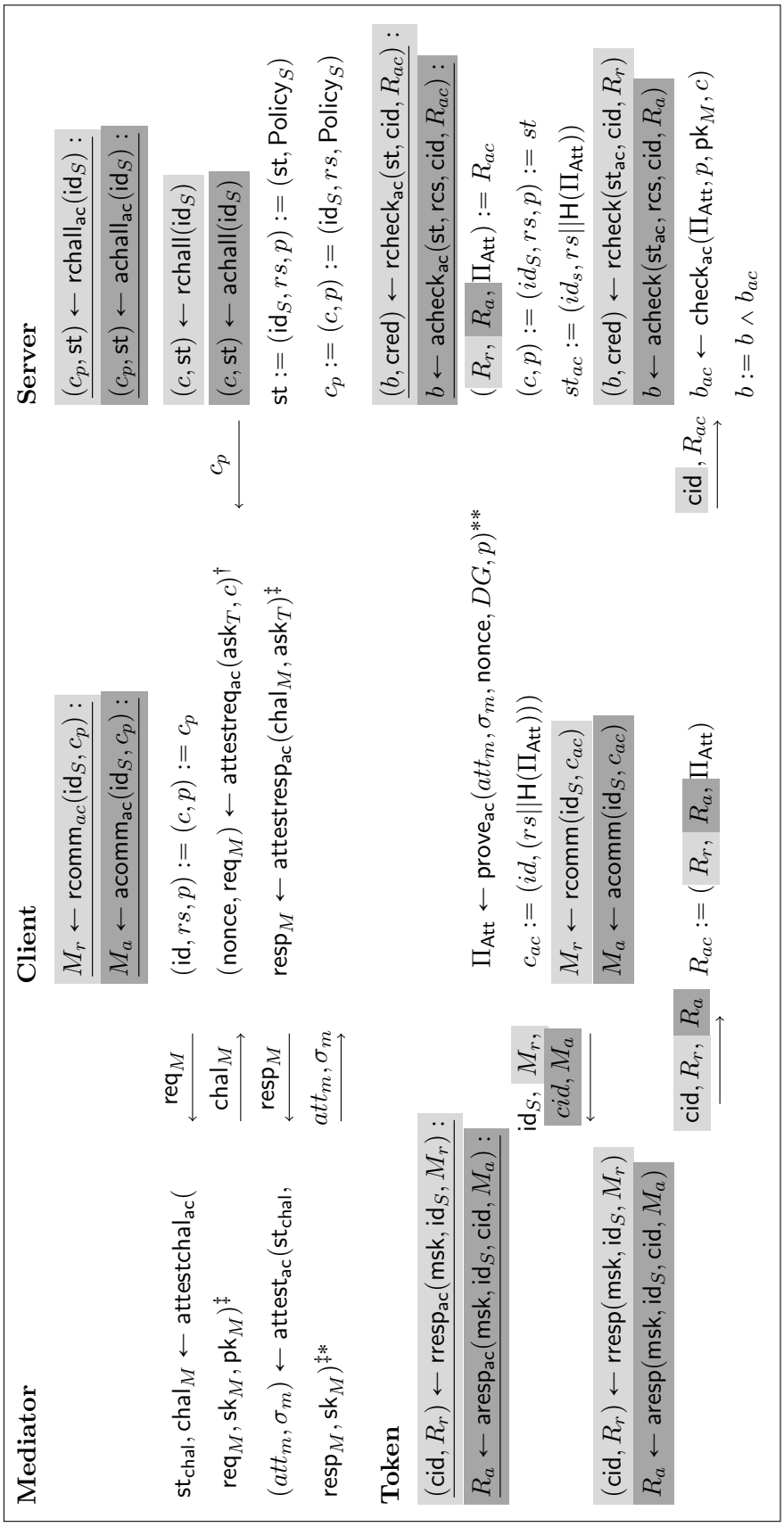


Figure 5.1: The FIDO-AC protocol for registration and authentication. The flow reuses definitions in [99] (Figure 1). ▫ - registration, ▪ - authentication, if not marked, applicable to both flows. † - eID read, ‡ - liveliness check, * - mediator attestation, ** - ZKP generation.

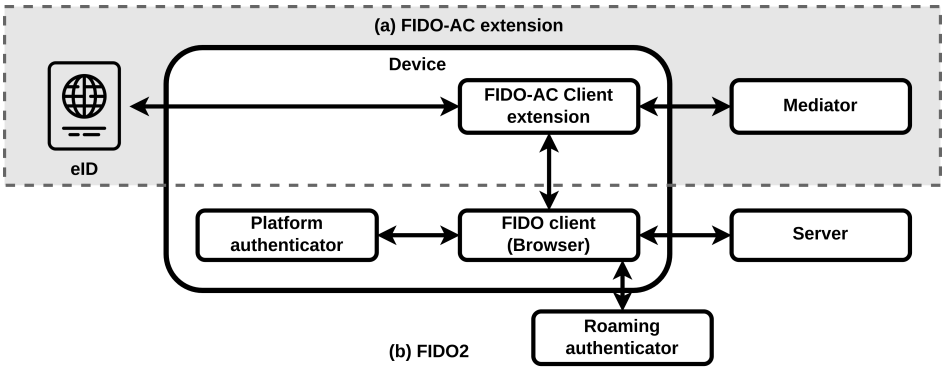


FIGURE 5.2: Differences between (a) FIDO-AC and (b) FIDO2. Additional parts of FIDO-AC are given in the gray box.

comprises three subsystems, namely anonymous credentials (AC), mediator, and FIDO extension integrated to provide the needed functionalities. The AC module will gather and supply all the necessary information required to fully realize an AC system while utilizing the binding to FIDO as the medium to deliver the information. Compared to using the standard FIDO2 protocol, users have to install an additional FIDO-AC application and scan their eID for proof of interaction. The application is also used to arbitrarily disclose the attributes of users, using a non-interactive zero-knowledge proof system. Note that installation is a one-time setup, while the result from the scanning of eID can be reused if the user explicitly permits caching. Additionally, the FIDO-AC framework relies on a trusted mediator to produce the proof of interaction with an eID for each FIDO2 transaction.

Below we describe the FIDO-AC protocol (illustrated in Figure 5.1) using the notion of passwordless authentication with attributes and mediator we introduced in the previous section. The description will use the standard FIDO passwordless authentication as a building block. Thus, we will use the notion provided by Hanzlik et al.[99]: rchall, achall, rcomm, acomm, rresp, aresp, rcheck, acheck, to denote the standard protocol and use the suffix *ac* (e.g., $\mathsf{rchall}_{\mathsf{ac}}$) to indicate our PAwAM. Please note that only $\mathsf{rchall}_{\mathsf{ac}}$, $\mathsf{achall}_{\mathsf{ac}}$, $\mathsf{rcomm}_{\mathsf{ac}}$, $\mathsf{acomm}_{\mathsf{ac}}$, and $\mathsf{rcheck}_{\mathsf{ac}}$, $\mathsf{acheck}_{\mathsf{ac}}$ introduce additional steps to the FIDO protocol, which remain the same for registration and authentication processes. Therefore, we only provide descriptions of the functions mentioned above

and the pseudocode for the new algorithms (see Table 5.1).

The $\mathsf{rcomm}_{\mathsf{ac}}$ and $\mathsf{acomm}_{\mathsf{ac}}$ algorithms are triggered with server identity id_S and challenge c_p (the standard FIDO challenge is extended with a policy). Then, the client extracts data from the eID (see step † in Figure 5.1 and Section 5.5.2), which is followed by a communication with the mediator to run the liveliness check (steps ‡) and an attestation generation (step *). Then, the client runs the zero-knowledge proof (ZKP) generation (step **). The attestation is hashed and appended to the challenge, which is passed to the rcomm or acomm functions. Regarding the $\mathsf{rcheck}_{\mathsf{ac}}$ and $\mathsf{acheck}_{\mathsf{ac}}$ functions, we modify the generated challenge with the hashed attestation values and ZKP. Then, we use the modified challenge in the rcheck and acheck algorithms. Finally, we run a $\mathsf{check}_{\mathsf{ac}}$ algorithm, which verifies both ZKP and attestation. The server's decision depends on both FIDO and FIDO-AC checks.

5.5.2 Anonymous Credentials

FIDO-AC can use any anonymous credential system supporting an active non-shareability test. In literature, this is usually ensured via binding the secret key for the credentials to a hardware token. Due to the construction of the FIDO-AC framework, it can not only support any attribute system with a non-shareability property but also improve the privacy guarantees of the final solution. Without loss of generality, we will use ICAO-compliant eIDs as the basis for the credential part of FIDO-AC. The way we will use the eID can be described as follows. A FIDO-AC-specific application will extract the authenticated data for the eID. A mediator can verify the data using the Passive Authentication protocol described in Section 5.2.1, and then run the liveliness check.

The liveliness check, used by mediator, verifies whether the user owns the provided authenticated data. Such a check is implemented in the ICAO-based eID infrastructure using Active Authentication or its variant that we will use namely Chip Authentication (CA). The idea behind those authentication methods is that eID is equipped with a secret key stored in secure memory. In the eID setting, it is assumed that the key

never leaves this secure memory and cannot be extracted. During CA, the eID proves knowledge of the secret key with respect to a public key bound to the authenticated data.

We will now summarize the liveliness test in more detail. The test starts with the eID sending the hash value of the data it stores, including the public key and the issuer's signature (see Section 5.2.1 for more information), as well as the relying party's challenge and application's nonce for binding purposes. Using this data, the mediator performs both the CA and PA. After the session keys are replaced, the mediator queries the eID for a random challenge for the Terminal Authentication (TA) protocol. This command is encrypted using the session keys that are the result of the CA protocol. Thus, it implicitly prove knowledge of the secret key corresponding to the disclosed eID public key, which is bound to authenticated data. The result of the liveliness test is a signature of the mediator attesting that it performed the liveliness test for the signed data. The liveliness test is captured in the algorithms ($\mathsf{attestreq}_{\mathsf{ac}}$,$\mathsf{attestchal}_{\mathsf{ac}}$,$\mathsf{attestresp}_{\mathsf{ac}}$,$\mathsf{attest}_{\mathsf{ac}}$), and the pseudocode for the algorithms can be found in Table 5.1.

Disclosing Attributes, in the FIDO-AC system, is implemented using zero-knowledge (ZK) proofs. Recall that the mediator's signature is under a specific hash value and the relying party's challenge. This hash value is a salted hash of the hashed attributes of the user data and, as such, contains data that should not be disclosed to the verifier. At the same time, the verifier must check this double-hashed salted data to adhere to some policies. Therefore, we will use a proof system to do exactly this.

To limit communication, we consider using the non-interactive zero-knowledge proof (NIZK) system, as it do not require interaction between the prover and the verifier. There exist many non-interactive ZK-proof systems supporting arbitrary computation to be proven, namely Groth'16 ZK-SNARK [92], the setup less bulletproof [41], and ZK-STARK [35]. The exact choice of the proof system is not pertinent to the design and is left as the implementation details. The anonymous credential in FIDO-AC will take the form of a non-interactive zero-knowledge proof about some properties of the data from the eID, which are represented by a double-hashed salted value. The

credential will also include the attestation of the mediator in the form of a standard digital signature on this hash value that will be returned as the result of the liveliness test. The ZK proof of attributes is captured in the algorithms ($\mathsf{prove}_{\mathsf{ac}}$,$\mathsf{check}_{\mathsf{ac}}$), and the pseudocode for the algorithms can be found in Table 5.1.

5.5.3 FIDO-AC extension

FIDO2 extensions are the recommended way to extend FIDO2 functionality. Therefore our design follows the best practices and introduces a new extension called *fidoac*. However, including a new extension to the FIDO2 messages is not enough to fully integrate with the FIDO2 ecosystem. Our objective is to minimize the effort of adapting the FIDO-AC system and provide a smooth integration with existing FIDO clients and authenticators. Therefore, we analyzed the available solutions to select the most suitable approach for the FIDO-AC system. We identified three approaches: custom client[93], relying party modification[146], and client extension[158] (detailed evaluation in Appendix C.4). However, none of the above-mentioned solutions is sufficient to design an architecture that fulfills our requirements. Therefore, we followed a hybrid approach based on the modifications introduced before and after WebAuthn API is called, and marginal modification in the FIDO server. First, the FIDO assertion request needs to be extended to include *fidoac* extension. Second, an additional JavaScript (*fidoac.js*) needs to be included in the page, which handles the *navigator.credentials.get* execution. Finally, the FIDO server needs an additional code snippet used to verify the FIDO assertion with the *fidoac* extension. FIDO-AC modified the challenge used, and a detailed discussion of the modification can be found in Appendix C.3.

Our approach is fully compatible with FIDO2 (**R.3**) and does not modify user-facing FIDO2 parties (**R.5**). Thus, it can be considered to be as scalable as a pure FIDO2 system (**R.6**). The modifications of the FIDO server are trivial to implement with the existing FIDO2 libraries (e.g., custom extensions in SimpleWebAuthn Server[¶]). Similarly, the verification modification requires only two widely supported primitives (SHA-256 and base64 encoding). Regarding *fidoac.js*, the JavaScript can be imported

[¶] https://www.npmjs.com/package/@simplewebauthn/server

directly from an external source (e.g., hosted by FIDO-AC) to the web page. Therefore, we claim that the introduced modifications fulfil requirement **R.5**.

5.6 Security Analysis

We will now state the security guarantees of FIDO-AC formally. However, first we will recall standard cryptographic definitions for signature scheme and zero-knowledge proof systems.

5.6.1 Signatures and Proof System

Definition 16 *A signature scheme σ consists of PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *with the following syntax.*

$\mathsf{KeyGen}(\lambda)$**:** *This non-deterministic algorithm takes as input the security parameter λ and outputs a pair of verification and signing keys* $(\mathsf{vk}, \mathsf{sk})$.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{m})$**:** *This algorithm takes as input a signing key* sk *and a message* m *and outputs a signature σ.*

$\mathsf{Verify}(\mathsf{vk}, \mathsf{m}, \sigma)$**:** *This deterministic algorithm takes as input a verification key* vk*, a message* m*, and a signature σ and outputs either* 0 *or* 1.

We define the following properties of a signature scheme.

Correctness: *For every security parameter $\lambda \in \mathbb{N}$ and every message $\mathsf{m} \in \{0,1\}^*$, that given $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda)$, $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{m})$, it holds that $\mathsf{Verify}(\mathsf{vk}, \mathsf{m}, \sigma) = 1$.*

Existential Unforgeability under Chosen Message Attacks: *Let $\lambda \in \mathbb{N}$ be a security parameter. We define the advantage $\mathsf{EUF\text{-}CMA}^{\mathcal{A}}(\lambda)$ of an adversary $\mathcal{A}$ against unforgeability under chosen message attack as the following probability:*

$$\Pr\left[\begin{array}{l|l} \mathsf{Verify}(\mathsf{vk},\mathsf{m}^*,\sigma^*)=1 \ : & (\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{Setup}(\mathsf{sk}); \\ & (\sigma^*,\mathsf{m}^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)} \end{array}\right]$$

where m^* *was not queried to the* $\mathsf{Sign}(\mathsf{sk},\cdot)$ *oracle, and the probability is taken over the random coins of* Setup *and the random coins of* $\mathcal{A}$.

Definition 17 (Non-Interactive Argument System) *Let* $\mathcal{R}$ *be an* **NP***-relation and* $\mathcal{L}_{\mathcal{R}}$ *be the language defined by* $\mathcal{R}$. *A non-interactive argument for* $\mathcal{L}_{\mathcal{R}}$ *consists of algorithms* (Setup, Prove, Verify) *with the following syntax.*

$\mathsf{ZKSetup}(\lambda)$**:** *Takes as input a security parameter* λ*, and outputs a common reference string* crs.

$\mathsf{ZKProve}(\mathsf{crs},\mathsf{stmt},\mathsf{wit})$**:** *Takes as input a common reference string* crs*, a statement* stmt *and a witness* wit*, and outputs either a proof* π_{zkp} *or* $\perp$.

$\mathsf{ZKVer}(\mathsf{crs},\mathsf{stmt},\pi_{\mathsf{zkp}})$**:** *Takes as input the common reference string* crs*, a statement* stmt*, and a proof* π_{zkp}*, and outputs either* 0 *or* 1.

Completeness: *For all security parameters* $\lambda \in \mathbb{N}$*, all statements* $\mathsf{stmt} \in \mathcal{L}_{\mathcal{R}}$ *and all witnesses* wit *with* $\mathcal{R}(\mathsf{stmt},\mathsf{wit})=1$, $\mathsf{crs} \leftarrow \mathsf{ZKSetup}(\lambda)$,
and $\pi_{\mathsf{zkp}} \leftarrow \mathsf{ZKProve}(\mathsf{crs},\mathsf{stmt},\mathsf{wit})$*, it holds that* $\mathsf{ZKVer}(\mathsf{crs},\mathsf{stmt},\pi_{\mathsf{zkp}})=1$.

Computational Soundness: *We define the advantage* $\mathsf{Sound}^{\mathcal{A}}(1^\lambda)$ *of an adversary* $\mathcal{A}$ *in breaking the soundness of the proof system as the probability defined below, where the probability is taken over the random coins of* ZKVer.

$$\Pr\left[\begin{array}{c} \mathsf{ZKVer}(\mathsf{crs},\mathsf{stmt},\pi_{\mathsf{zkp}})=1 \ : \\ \\ \mathsf{crs}\leftarrow\mathsf{ZKSetup}(\lambda); \\ (\pi_{\mathsf{zkp}},\mathsf{stmt})\leftarrow\mathcal{A}^{\mathsf{Verify}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}); \\ \mathsf{stmt}\notin\mathcal{L}_{\mathcal{R}} \end{array}\right]$$

Zero-Knowledge: *We define the advantage* $\mathsf{ZK}^{\mathcal{A}}(\lambda)$ *of an adversary* $\mathcal{A}$ *in breaking the zero-knowledge property of the proof system as the probability defined below, where the probability is taken over random coins of* ZKProve. *We say that the proof system is zero-knowledge if there exist simulator algorithms* $(\mathcal{SIM}_1,\mathcal{SIM}_2)$ *such that for all PPT adversaries* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that* $\mathsf{ZK}^{\mathcal{A}}(\lambda)\leqslant\mathsf{negl}(\lambda)$.

$$\left|\Pr\left[\begin{array}{c} \mathcal{A}(\pi_{\mathsf{zkp}}{}^*)=1 \ : \\ \\ \\ \end{array}\begin{array}{c} \mathsf{crs}\leftarrow\mathsf{ZKSetup}(\lambda); \\ (\mathsf{stmt},\mathsf{wit})\leftarrow\mathcal{A}(\mathsf{crs}); \\ \mathcal{R}(\mathsf{stmt},\mathsf{wit})=1; \\ \pi_{\mathsf{zkp}}{}^*\leftarrow\mathsf{ZKProve}(\mathsf{crs}, \\ \mathsf{stmt},\mathsf{wit}) \end{array}\right]-\right.$$
$$\left.\Pr\left[\begin{array}{c} \mathcal{A}(\pi_{\mathsf{zkp}}{}^*)=1 \ : \\ \\ \\ \end{array}\begin{array}{c} (\mathsf{crs},\mathsf{st})\leftarrow\mathcal{SIM}_1(\lambda); \\ (\mathsf{stmt},\mathsf{wit})\leftarrow\mathcal{A}(\mathsf{crs}); \\ \mathcal{R}(\mathsf{stmt},\mathsf{wit})=1; \\ \pi_{\mathsf{zkp}}{}^*\leftarrow\mathcal{SIM}_2(\mathsf{crs},\mathsf{stmt},\mathsf{st}) \end{array}\right]\right|$$

5.6.2 Assumptions

Assumption 1 (Data Groups) *Let* DG_0 *and* DG_1 *be the full data groups specified in [104], which contain personal data with attributes stored in the data group 1 file EF.DG1 (see Section 4.7.1.1 in [104]). As per [104][Section 4.7.1.1], in addition to*

the respective personal attributes Att_0 *and* Att_1*, the data groups 1 in* DG_0 *and* DG_1 *contain additional high entropy data. Thus, we can define alternative versions of* DG_0 *and* DG_1*, denoted by* DG'_0 *and* DG'_1*, with keys* $\mathsf{ask}'_0, \mathsf{ask}'_1$*. Except for the personal attributes* Att_0 *and* Att_1*, we assume that those data groups are different and capture this formally below.*

We will denote the advantage $\mathsf{Assu}^{\mathcal{A}}_{DG}(\lambda)$ *of an adversary* $\mathcal{A}$ *in breaking this assumption as the following probability:*

$$\Big|\Pr\left[\bar{b} = b \;:\; \begin{array}{c} (\mathsf{Att}_0, \mathsf{Att}_1, \mathsf{st}) \leftarrow \mathcal{A}(\lambda); \\ \mathsf{ask}_0 \leftarrow \mathsf{IssCred}(\mathsf{Att}_0); \\ \mathsf{ask}_1 \leftarrow \mathsf{IssCred}(\mathsf{Att}_1); \\ b \xleftarrow{\$} \{0,1\}; \\ D := (\mathsf{pk}_{eID}, \pi_{PA}, \mathsf{ask}'_b, \mathsf{H}(DG'_b)); \\ \bar{b} \leftarrow \mathcal{A}(\mathsf{st}, \{(\mathsf{ask}_i, DG_i)\}_{i\in\{0,1\}}, D) \end{array}\right] - \frac{1}{2}\Big|,$$

where the probability is taken over the random choice of $\mathcal{A}$*. We assume this assumption holds if its advantage is negligible for any PPT adversary* $\mathcal{A}$*.*

The value D contains a public key pk_{eID}, and the corresponding secret key ask'_b, which are not bound to personal attributes. π_{PA} is a signature under $\mathsf{H}(DG'_b)$, also not leaking anything. The only value in D that contains personal information is $\mathsf{H}(DG'_b)$. According to [104], there are no less than 35 data values independent of personal data and only corresponding to the document (elements 03, 05, 12). Each value can be one of 37 possibilities ([A-Z], [0-9], '<'), which gives around 182 random bits. Thus, in the random oracle model, this assumption is reasonable. Note that we did not count additional values like the expiration date and checksum.

Assumption 2 (Passive Authentication) *Let us denote by* $DG(\mathsf{Att})$ *the data group encoding of attributes* Att *and by* $(\mathsf{sk}_M, \mathsf{pk}_M)$ *the Mediator keypair. We will denote the advantage* $\mathsf{Assu}^{\mathcal{A}}_{PA}(\lambda)$ *of an adversary* $\mathcal{A}$ *in breaking this assumption as the following*

probability:

$$\Pr\left[\begin{array}{c} b_{PA}=1 \end{array} : \begin{array}{c} (\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}) \leftarrow \mathcal{A}(\lambda); \\ b_{PA} \leftarrow PA_{verify}(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}) \end{array}\right],$$

where the probability is taken over the random choice of $\mathcal{A}$. We assume this assumption holds if its advantage is negligible for any PPT adversary $\mathcal{A}$.

Assumption 3 (Liveliness/Chip Authentication) *Let us denote by $DG(\mathsf{Att})$ the data group encoding of attributes Att, by $(\mathsf{sk}_M, \mathsf{pk}_M)$ the Mediator keypair, and by cmd the get random nonce command of Terminal Authentication. We also introduce the oracle $\mathsf{IssCred}'(\cdot)$ that takes as input Att and outputs $DG(\mathsf{Att}), \mathsf{pk}_{eID}, \pi_{PA}$, such that $1 \leftarrow PA_{verify}(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA})$. It also keeps all the output values in a set Iss. This allows the adversary to create up to m tokens/users. Additionally, we give the adversary access to the oracle Med that takes as input index $i \in \{1, \ldots, m\}$ and challenge cmd_{cha} and outputs the Mediator response where $\mathsf{resp}_M \leftarrow CA(\mathsf{pk}_M, cmd_{cha}, \mathsf{ask}_{T_i})$. The oracle keeps the queried inputs cmd_{cha} on a list L_M. We will denote the advantage $\mathsf{Assu}^{\mathcal{A}}_{CA}(\lambda)$ of an adversary $\mathcal{A}$ in breaking this assumption as the following probability:*

$$\Pr\left[\begin{array}{l} b_{CA}=1 \\ \\ \wedge \\ \\ cmd_{cha} \\ \notin L_M \end{array} : \begin{array}{c} (\mathsf{pk}_{eID}, \mathsf{st}) \leftarrow \mathcal{A}^{\mathsf{Med}(\cdot,\cdot),\mathsf{IssCred}'(\cdot)}(\lambda); \\ key_{ses} \leftarrow KE(\mathsf{pk}_{eID}, \mathsf{sk}_M); \\ cmd_{cha} \leftarrow AE\text{-}ENC(key_{ses}, cmd); \\ (\mathsf{H}(DG), \pi_{PA}, \mathsf{resp}_M) \leftarrow \mathcal{A}(\mathsf{st}, cmd_{cha}); \\ b_{PA} \leftarrow PA_{verify}(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}); \\ b_{CA} \leftarrow CA_{Verify}(\mathsf{resp}_M, key_{ses}) \end{array}\right],$$

where the probability is taken over the random choice of $\mathcal{A}$. We assume this assumption holds if its advantage is negligible for any PPT adversary $\mathcal{A}$.

Both assumptions hold, as shown in the security analysis of the extended access control protocol [57]. In more detail, Assumption 2 holds since the signature scheme used to sign the data groups is unforgeable. Assumption 3 holds since to create a valid

response for the Mediator challenge without the secret key corresponding to pk_{eID}, one would have to know the session key between the Chip and the Terminal after the Chip Authentication protocol. As shown in [57], this holds.

5.6.3 Impersonation Security

Theorem 1 *Let us define the advantage of an adversary $\mathcal{A}$ in winning the experiment in Definition 10 as:*

$$\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Imp},\mathsf{PAwAM}} := \Pr[\mathbf{Imp}^{\mathcal{A}}_{\mathsf{PAwAM}} = 1].$$

Similarly, we denote $\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Imp},\mathsf{PLA}}$ the advantage of an adversary in winning the impersonation experiment for a passwordless authentication schemes as defined in [99].
For the FIDO-AC (Figure 5.1) we have:

$$\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Imp},\mathsf{PAwAM}} = \mathbf{Adv}^{\mathcal{A}}_{\mathbf{Imp},\mathsf{PLA}}.$$

Informally, FIDO-AC (Figure 5.1) is secure against impersonation as long as the underlying FIDO scheme is secure against impersonation and supports extension fields.

Let us assume that an adversary $\mathcal{A}$ exists against the FIDO-AC scheme presented in Figure 5.1. Without loss of generality, we assume that this adversary always set ups the system so that all tokens can pass the server's policies. We can do this since, in the case of impersonation security, the attributes do not condition the adversary's winning conditions. This assumption only maximizes $\mathcal{A}$'s success probability and simplifies our considerations.

We show this theorem by constructing a reduction algorithm $\mathcal{R}$ that will use $\mathcal{A}$ as a sub-procedure to break the impersonation resistance of the underlying passwordless authentication scheme PLA used as a building block. The reduction plays the role of the adversary in the PLA impersonation experiment while simultaneously playing the role of the challenger in the PAwAM impersonation experiment. We describe how the reduction answers all queries for the adversary $\mathcal{A}$.

The reduction $\mathcal{R}$ works as follows:

- **Setup.** During the setup phase, the reduction runs the setup phase for the PLA scheme and keeps the attributes and policies specified by $\mathcal{A}$ in lists $(\mathsf{Policy}_{S_1}, ..., \mathsf{Policy}_{S_n}) := \mathsf{Policy}_{LS}$ and $(\mathsf{Att}_{T_1}, ..., \mathsf{Att}_{T_m}) := \mathsf{Att}_{LT}$. For each $i \in \{1, \ldots, m\}$ the reduction also executes $\mathsf{ask}_{T_i} \leftarrow \mathsf{IssCred}(\mathsf{Att}_{T_i})$, while retaining all the keys. The reduction also generates the Mediator's keypair $(\mathsf{sk}_M, \mathsf{pk}_M)$ honestly, allowing it to attest to attributes for all tokens.

- **Online Phase.** During the online phase the reduction answers all Mediator related oracle queries (i.e., oracles MedReq, MedChal, MedResp, resp_M, MedAttest) according to the FIDO-AC protocol. Note that this is possible, since it knows the secret key sk_M and the attributes of tokens and server policies. $\mathcal{R}$ behaves a bit differently in case of the Start, Challenge and Complete oracles.
  - To answer a $\mathsf{Start}(\pi_S^{i,j})$ query made by $\mathcal{A}$ the reduction makes the same call to the PLA challenger, receiving challenge c. It then output the challenge with policy $c_p := (c, \mathsf{Policy}_{S_i})$.
  - To answer a $\mathsf{Challenge}(\pi_T^{i,j}, \mathsf{id}_S, \mathsf{cid}, M)$ query made by the adversary, the reduction makes the same call to the PLA challenger.
  - To answer a $\mathsf{Complete}(\pi_S^{i,j}, \mathsf{cid}, R_{ac})$ query the reduction first parses R_{ac} as (R, Π_{Att}) and then calls the $\mathsf{Complete}(\pi_S^{i,j}, \mathsf{cid}, R)$ oracle provided by the PLA challenger.

- **Output Phase.** Finally, adversary $\mathcal{A}$ terminates.

Since we assume that $\mathcal{A}$ wins the impersonation experiment, there must exist a server handle $\pi_S^{i,j}$ that fulfills all the conditions for the impersonation experiments. Note that those conditions are the same as in the case of PLA. It remains to show that the change of FIDO-AC did not influence the acheck result in PLA. The only difference is that the message M in all calls of the adversary $\mathcal{A}$ to the Challenge oracle contains the hash value $\mathsf{H}(\Pi_{\mathsf{Att}})$ additionally. However, as assumed, this does not change the output of the acheck algorithm since we assumed that challenges can be extended.

Thus, if $\mathcal{A}$ successfully breaks the security against impersonation, then with the same probability, the reduction $\mathcal{R}$ can break the security against impersonation for the PLA scheme.

5.6.4 Unlinkability

Theorem 2 *For $x \in \{\mathbf{w}, \mathbf{m}, \mathbf{s}\}$ let us define the advantage of an adversary $\mathcal{A}$ in winning the experiment in Definition 12 as:*

$$\mathbf{Adv}^{\mathcal{A}}_{x\mathbf{Unl},\mathsf{PAwAM}} := |\Pr[x\mathbf{Unl}^{\mathcal{A}}_{\mathsf{PAwAM}} = 1] - 1/2|.$$

Similarly, we denote by $\mathbf{Adv}^{\mathcal{A}}_{x\mathbf{Unl},\mathsf{PLA}}$ the advantage of an adversary in winning the impersonation experiment for a passwordless authentication scheme as defined in [99]. Additionally, q_h, q_L, q_R, respectively, denote the number of random oracle queries, queries to the Left *oracle, and queries to the* Right *oracle.*
In the random oracle model, for the FIDO-AC (Figure 5.1), we have:

$$\mathbf{Adv}^{\mathcal{A}}_{x\mathbf{Unl},\mathsf{PAwAM}} \leqslant +\mathbf{Adv}^{\mathcal{A}}_{x\mathbf{Unl},\mathsf{PLA}} + (q_L + q_R) \cdot (\frac{q_{\mathsf{H}}}{2^{\lambda}} + \mathsf{ZK}^{\mathcal{A}}).$$

Informally, FIDO-AC (Figure 5.1) is unlinkable as long as the underlying FIDO scheme is unlinkable and the proof system is zero-knowledge.

We will show this theorem via a series of small changes we make to experiment, which follows the well-known hybrid argument technique.

GAME$_0$ The original $x\mathbf{Unl}_{\mathsf{PAwAM}}$ experiment.

GAME$_1$ Let $(\mathcal{SIM}_1, \mathcal{SIM}_2)$ be the simulator for the zero-knowledge experiment of the underlying proof system. We will now define $(q_L + q_R)$ sub-games, where we keep a counter q_{LR} for the number of current queries to either Left or Right. In the i-th sub-game, instead of computing the proof π_{zkp} as $\mathsf{ZKProve}(\mathsf{crs}, (m, \mathsf{Policy}_S), (DG, \mathsf{nonce}))$ for the i-th query (i.e., when during the experiment the query

counter q_{LR} is i) we use the simulator $\mathcal{SIM}_2(\mathsf{crs}, (m, \mathsf{Policy}_S), \mathsf{st})$, where $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathcal{SIM}_1(\lambda)$.

We increase the adversary's advantage by $\mathsf{ZK}^A(\lambda)$ with each sub-game change. Since there are $(q_L + q_R)$ queries, we have:

$$|\Pr[\mathbf{GAME}_{1,(q_L+q_R)}(\mathcal{A}) = 1] - \Pr[\mathbf{GAME}_0(\mathcal{A}) = 1]| \leqslant (q_L + q_R) \cdot \mathsf{ZK}^A(\lambda)$$

GAME$_2$ Let $\mathsf{nonce}_1, \ldots, \mathsf{nonce}_{q_L+q_R}$ be the nonce used to compute the attestation messages att_m in the $(q_L + q_R)$ queries to the Left and Right oracles. We abort the experiment in case the adversary makes a query of the form $(\cdot, \mathsf{nonce}_i)$, for any $i \in \{1, \ldots, (q_L + q_R)\}$, to the random oracle. Since the nonces are picked by the challenger in the unlinkability experiment, we can upper bound the probability of aborting by

$$|\Pr[\mathbf{GAME}_2(\mathcal{A}) = 1] - \Pr[\mathbf{GAME}_{1,(q_L+q_R)}(\mathcal{A}) = 1]| \leqslant \frac{q_\mathsf{H} \cdot (q_L + q_R)}{2^\lambda}$$

We now argue that the probability of the adversary $\mathcal{A}$ winning the unlinkability experiment in **GAME**$_2$ is the same as winning the unlinkability experiment for the underlying passwordless authentication PLA scheme. To this end, we will construct a reduction $\mathcal{R}$ that plays the role of the adversary against the PLA unlinkability experiment and the role of the challenger in this experiment for the PAwAM scheme. The reduction $\mathcal{R}$ works as follows:

- **Setup.** During the setup phase, the reduction runs the setup phase for the PLA scheme and keeps the attributes and policies specified by $\mathcal{A}$ in lists $(\mathsf{Policy}_{S_1}, ..., \mathsf{Policy}_{S_n}) := \mathsf{Policy}_{LS}$ and $(\mathsf{Att}_{T_1}, ..., \mathsf{Att}_{T_m}) := \mathsf{Att}_{LT}$. For each $i \in \{1, \ldots, m\}$ the

reduction also executes $\mathsf{ask}_{T_i} \leftarrow \mathsf{IssCred}(\mathsf{Att}_{T_i})$, while retaining all the keys. The reduction also generates the Mediator's keypair $(\mathsf{sk}_M, \mathsf{pk}_M)$ honestly, allowing it to attest to attributes for all tokens. It is worth noting that due to the changes made in $\mathbf{GAME}_1$, the reduction uses the $\mathcal{SIM}_1$ to generate the common reference string $(\mathsf{crs}, \mathsf{st}) \leftarrow \mathcal{SIM}_1(\lambda)$.

- **Phase 1.** The reduction translates all calls to the Start, Challenge, and Complete oracles to oracle calls for the PLA scheme and vice versa. This includes adding policies to PLA challenges and discarding the proof Π_{Att} from the responses to Complete. For more details, see the proof of Theorem 1. The reduction also honestly executes all the Mediator related oracles.

- **Phase 2.** $\mathcal{R}$ checks the conditions for the instance submitted by the adversary and submits them to the PLA challenger.

- **Phase 3.** The reduction runs the Phase 1 oracles as already described. In the case of oracles Left, Right, because of the changes made in $\mathbf{GAME}_1$, it is using the simulator $\mathcal{SIM}_2$ to create π_{zkp}. Other values in Π_{Att} are generated as prescribed, except att_m, which is picked uniformly at random. Note that the reduction knows the Mediator keys to create the attestation honestly.

- **Output Phase.** The adversary outputs bit $\hat{b}$, which is also the output of the reduction $\mathcal{R}$.

It remains to show that $\Pr[\mathbf{GAME}_2(\mathcal{A}) = 1] = \mathbf{Adv}^{\mathcal{R}}_{x\mathbf{Unl},\mathsf{PLA}}$. First, we notice that the only additional component in PAwAM that is the output of the Left and Right oracles, is the attestation $\Pi_{\mathsf{Att}} = (att_m, \sigma_m, \pi_{\mathsf{zkp}})$. Because for all proofs, we used $\mathcal{SIM}_2$, the proof π_{zkp} does not contain any information about bit $\hat{b}$. Moreover, in $\mathbf{GAME}_2$, we assume that $\mathcal{A}$ never queries the random oracle with $(\cdot, \mathsf{nonce}_i)$. It follows that att_m also does not give any information about bit $\hat{b}$. The bit $\bar{b}$ output by the adversary

$\mathcal{A}$ will allow the reduction $\mathcal{R}$ to win the unlinkability experiment for the PLA scheme since the same conditions apply.

5.6.5 Attribute Unforgeability

Theorem 3 *Let us define the advantage of an adversary $\mathcal{A}$ in winning the experiment in Definition 11 as:*

$$\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Att\text{-}Unf},\mathsf{PAwAM}} := \Pr[\mathbf{Att\text{-}Unf}^{\mathcal{A}}_{\mathsf{PAwAM}} = 1].$$

Furthermore, let us denote by q_M the number of supported queries to the MedChal *oracle. For the FIDO-AC (Figure 5.1) we have:*

$$\begin{aligned}\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Att\text{-}Unf},\mathsf{PAwAM}} = \mathsf{EUF\text{-}CMA}^{\mathcal{A}} + \mathsf{Sound}^{\mathcal{A}} \\ + \mathsf{Assu}^{\mathcal{A}}_{PA} + \frac{1}{q_M} \cdot \mathsf{Assu}^{\mathcal{A}}_{CA}.\end{aligned}$$

Informally, FIDO-AC (Figure 5.1) is secure against attribute unforgeability as long as the Mediator's signature cannot be forged, the eID protocols are secure, and the proof system cannot be used to prove false statements.

We begin the proof with the following observation. For the server to accept a response, we need the verification $\mathsf{check}_{\mathsf{ac}}(\Pi_{\mathsf{Att}}, p, pk_M, c)$ to output 1. This can only happen in case the proof π_{zkp} and signature σ_m are valid, where $\Pi_{\mathsf{Att}} = (att_m, \sigma_m, \pi_{\mathsf{zkp}})$.

We will show this theorem via a series of small changes we make to the experiment, which follows the well-known hybrid argument technique.

GAME$_0$ The original **Att-Unf**$_{\mathsf{PAwAM}}$ experiment. Let $\pi_S^{i,j}$ be the server instance that accepted the partnered session for which there does not exist a corresponding token $\pi_T^{i',j'}$ or $\pi_T^{i',j'}$ does not satisfy the policy of server id_S.

GAME$_1$ Same as **GAME**$_0$, but we abort in case the signature σ_m was never an output of the MedAttest algorithm.

Since the signature scheme used by the Mediator is existentially unforgeable, it follows that:

$$|\Pr[\mathbf{GAME}_1(\mathcal{A}) = 1] - \Pr[\mathbf{GAME}_0(\mathcal{A}) = 1]| \leqslant \mathsf{EUF\text{-}CMA}^{\mathcal{A}}.$$

The claim follows easily via a simple reduction to the unforgeability experiment. Note that the adversary in the attribute unforgeability experiment is only given the Mediator's public key pk_M. Hence, the reduction can use the public key given by the signature scheme challenger. The reduction can use the provided signing oracle to compute σ_m and answer Mediator oracle queries. The winning condition for the attribute unforgeability experiment states that the challenge c, which is also part of att_m, is not queried to a request. It follows that att_m is never queried by the reduction to the signature scheme signing oracle, which ends the claim.

GAME$_2$ Assume that $att_m := \mathsf{H}(\mathsf{H}(DG)||\mathsf{nonce})||c$ is the attestation message that is part of Π_{Att} which is accepted in $\mathsf{check}_{\mathsf{ac}}$ as defined above. Moreover, let Att be the attributes encoded in DG. The reduction can access DG since it sees all the random oracle queries. The reduction aborts in case $\mathsf{Policy}_S(\mathsf{Att}) = 0$.
We claim that:

$$|\Pr[\mathbf{GAME}_2(\mathcal{A}) = 1] - \Pr[\mathbf{GAME}_1(\mathcal{A}) = 1]| \leqslant \mathsf{Sound}^{\mathcal{A}}.$$

To prove the claim, we construct a reduction $\mathcal{R}$ that breaks the soundness property of the proof system. The reduction simulates the whole system honestly. Since we abort the experiment in case $\mathsf{Policy}_S(\mathsf{Att}) = 0$, it follows that π_{zkp} is a proof for a false statement. Thus, by returning $(\pi_{\mathsf{zkp}}, (m, \mathsf{Policy}_s))$, where $att_m = (m||c_m)$, the reduction will break the soundness property of the proof system.

With the above changes, the only way for the server verification $\mathsf{check}_{\mathsf{ac}}(\Pi_{\mathsf{Att}}, p, pk_M, c)$

to accept is for $\Pi_{\mathsf{Att}} = (att_m, \sigma_m, \pi_{\mathsf{zkp}})$ to include a sound proof π_{zkp} (i.e., for the data groups in att_m, we have that the attributes satisfy the policy) and a signature σ_m generated by the Mediator. Thus, the only way for the adversary is to:

- Create valid data groups DG containing public key pk_{eID} and π_{PA}, such that $PA_{verify}(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}) = 0$, and where the adversary knows the corresponding secret key ask. Informally, this captures the case of creating a new eID with fake data.

- Use existing data groups DG, public key pk_{eID} and π_{PA} without knowing the secret key ask, but create a valid resp_M to receive the Mediator's attestation. Informally, this captures the case of impersonating a valid user by passing the liveliness test / Chip Authentication step.

With the next game change, we will exclude the first case and later show that the probability of success in the last case is negligible. It is worth noting that in both cases the adversary must query the Mediator oracles to get a valid attestation.

GAME$_3$ The reduction aborts the experiment if amongst the queries to the $\mathsf{MedChal}$ oracle, there exists a request $\mathsf{req}_M = (\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}, c, \mathsf{nonce})$ for which the attributes Att corresponding to DG satisfy $\mathsf{Att} \notin \mathsf{Att}_{LT}$ and DG do not correspond to any honest token (i.e., DG contains the key pk_{eID}).
We claim that:

$$|\Pr[\mathbf{GAME}_3(\mathcal{A}) = 1] - \Pr[\mathbf{GAME}_2(\mathcal{A}) = 1]| \leqslant \mathsf{Assu}_{PA}^{\mathcal{A}}.$$

The claim follows directly from Assumption 2. We create a reduction $\mathcal{R}$ that will break this assumption. The reduction simulates the system honestly. Because of the previous changes, there must be a call to the $\mathsf{MedChal}$ oracle with input $\mathsf{req}_M = (\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA})$, where DG does not correspond to any attributes $\mathsf{Att} \notin \mathsf{Att}_{LT}$ or pk_{eID} is a public key that does not correspond to data groups for the token with attributes Att. Thus, the reduction can return $(\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA})$

and break Assumption 2. Informally, changing personal data (Att) or the full data groups (e.g., the public key pk_{eID}) is synonymous with creating a fake eID.

Now the only way for the adversary to win is to create a valid response to the Mediator's challenge for an existing token. Informally, for FIDO-AC (Figure 5.1) this means that the adversary passed the eID liveliness test for valid attributes $\mathsf{Att} \in \mathsf{Att}_{LT}$ for which $\mathsf{Policy}(\mathsf{Att}) = 1$. Thus, we will now show that $|\Pr[\mathbf{GAME}_3(\mathcal{A}) = 1] \leqslant \frac{1}{q_M} \cdot \mathsf{Assu}^{\mathcal{A}}_{CA}$. We prove this claim by constructing a reduction $\mathcal{R}$ that uses adversary $\mathcal{A}$ in $\mathbf{GAME}_3$ to break Assumption 3. The reduction simulates the system honestly, except that it uses the $\mathsf{Med}, \mathsf{IssCred}'$ oracles provided by a challenger in Assumption 3 to answer queries to the Mediator oracles. The reduction picks one query to the MedChal oracle (by randomly picking an index i from $\{1, \ldots, q_M\}$), and for that query, it proceeds as follows. It extracts pk_{eID} from $\mathsf{req}_M = (\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}, c, \mathsf{nonce})$ and submits it to the challenger of Assumption 3. In return, it receives cmd_{cha} and it creates the challenge to $\mathcal{A}$ as $\mathsf{chal}_M := (\mathsf{pk}_M, cmd_{cha})$. The reduction then finds the corresponding response resp_M amongst the calls to oracle MedAttest. Because of the winning conditions, the adversary $\mathcal{A}$ never queries the MedResp oracle with chal_M. So the reduction never queries the Med oracle provided by the Assumption 3 challenger (i.e., one of the winning conditions is $cmd_{cha} \notin L_M$). Note that this only happens if the reduction chooses the correct index i, which happens with probability $\frac{1}{q_M}$. However, if that happens, the reduction can return $(\mathsf{H}(DG), \pi_{PA}, \mathsf{resp}_M)$ and break Assumption 3. With this, we proved the claim and the theorem.

5.6.6 Origin Privacy

Theorem 4 *Let us define the advantage of an adversary $\mathcal{A}$ in winning the experiment in Definition 13 as:*

$$\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Orig\text{-}Priv}, \mathsf{PAwAM}} := |\Pr[\mathbf{Orig\text{-}Priv}^{\mathcal{A}}_{\mathsf{PAwAM}} = 1] - 1/2|.$$

For the FIDO-AC (Figure 5.1) we have:

$$\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Orig\text{-}Priv},\mathsf{PAwAM}} = 0.$$

Informally, FIDO-AC (Figure 5.1) provides perfect origin privacy without relying on any assumptions.

The only information that depends on the bit b that is provided to an adversary $\mathcal{A}$ against origin privacy is the request

$$\mathsf{req}_{M,b} = (\mathsf{H}(DG), \mathsf{pk}_{eID}, \pi_{PA}, c, \mathsf{nonce}), \text{ and}$$

response

$$\mathsf{resp}_{M,b} \leftarrow CA(\mathsf{pk}_M, cmd_{cha}, \mathsf{ask}_T).$$

The response $\mathsf{resp}_{M,b}$ is independent of any output of the server, which only generates the challenge c. Thus, only the request req_M depends on an output created by server S_b. However, since c is a unique challenge (i.e., a uniformly random string of bits), it follows that it does not leak any information about bit b.

5.6.7 One-time Attribute Privacy

Theorem 5 *Let us define the advantage of an adversary $\mathcal{A}$ in winning the experiment in Definition 14 as:*

$$\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Att\text{-}Priv},\mathsf{PAwAM}} := |\Pr[\mathbf{Att\text{-}Priv}^{\mathcal{A}}_{\mathsf{PAwAM}} = 1] - 1/2|.$$

Let denote by m the number of tokens in the system. For the FIDO-AC (Figure 5.1) we have:

$$\mathbf{Adv}^{\mathcal{A}}_{\mathbf{Att\text{-}Priv},\mathsf{PAwAM}} = \frac{m^2}{2} \cdot \mathsf{Assu}^{\mathcal{A}}_{DG}.$$

Informally, FIDO-AC (Figure 5.1) provides one-time attribute privacy as long as the hashes of data groups of different documents but for the same personal data are indistinguishable.

We will prove the theorem by constructing a reduction $\mathcal{R}$ using an adversary $\mathcal{A}$ against the one-time attribute privacy experiment to break Assumption 1.

- **Setup.** During the setup phase, the reduction runs the setup phase for the PLA scheme and keeps the attributes and policies specified by $\mathcal{A}$ in lists $(\mathsf{Policy}_{S_1}, ..., \mathsf{Policy}_{S_n}) := \mathsf{Policy}_{LS}$ and $(\mathsf{Att}_{T_1}, ..., \mathsf{Att}_{T_m}) := \mathsf{Att}_{LT}$. It then picks two distinct indexes $j_0, j_1 \xleftarrow{\$} \{1, \ldots, m\}$ and for each $i \in \{1, \ldots, m\} \backslash \{j_0, j_1\}$ the reduction also executes $\mathsf{ask}_{T_i} \leftarrow \mathsf{IssCred}(\mathsf{Att}_{T_i})$, while retaining all the keys. The reduction now sends $(\mathsf{Att}_{T_0}, \mathsf{Att}_{T_1})$ to the challenger of Assumption 1, receiving back $(\mathsf{ask}_{j_0}, DG_{j_0}), (\mathsf{ask}_{j_1}, DG_{j_1}), (\mathsf{pk}_{eID}, \pi_{PA}, \mathsf{ask}'_b, \mathsf{H}(DG'_b)))$.

- **Phase 1.** The reduction honestly answers all the queries. It is worth noting that it possesses all the required information, including the attribute secret keys $\mathsf{ask}_{j_0}, \mathsf{ask}_{j_1}$ and data groups DG_{j_0}, DG_{j_1} for the tokens j_0 and j_1.

- **Challenge Phase 1.** The adversary $\mathcal{A}$ outputs two token identifiers T_0, T_1 and server identifier S. The reduction aborts in case it did not guess the correct tokens. The reduction proceeds if $T_0 = T_{j_0}$, $T_1 = T_{j_1}$ or $T_0 = T_{j_1}$, $T_1 = T_{j_0}$. We will show the reduction for the first case, while the second only increases the reduction of not aborting by 2. In other words, the probability that the reduction aborts is smaller than $2 \cdot \frac{1}{m^2}$.

- **Challenge Phase 2.** The reduction does not refresh the keys but will use $(\mathsf{ask}'_b, \mathsf{H}(DG'_b))$ in the next phase instead.

- **Challenge Phase 3.** First, the reduction runs $(c, \cdot) \leftarrow \mathsf{achall}_{\mathsf{ac}}(\mathsf{id}_S)$ and samples a random nonce $\mathsf{nonce} \xleftarrow{\$} \{0,1\}^\lambda$. It then sets $\mathsf{req}_{M,b} := (\mathsf{H}(DG'_b), \mathsf{pk}_{eID}, \pi_{PA}, c, \mathsf{nonce})$ and sends it to the adversary that responds with $\mathsf{chal}_{M,b}$. The reduction parses

the challenge as $(\mathsf{pk}_M, cmd_{cha})$ and executes $\mathsf{resp}_{M,b} \leftarrow CA(\mathsf{pk}_M, cmd_{cha}, \mathsf{ask}'_b)$.

- **Output Phase.** Finally, the adversary outputs $\bar{b}$, which is also the output of the reduction $\mathcal{R}$.

It is easy to see that if adversary $\mathcal{A}$ wins the one-time attribute privacy experiment, then the reduction $\mathcal{R}$ will break Assumption 1. Thus, this ends the proof.

5.6.8 Mediator Threat Analysis

The Mediator in FIDO-AC can be instantiated in multiple ways. We first rule out the possibility of delegating the mediator role to the client's browser, as it lacks a trusted execution environment (i.e., the verifier cannot trust such a mediator), and the configuration of the relying party acting as a mediator, as it breaks the privacy assumptions (i.e., linking user using identifiable properties of eID). Therefore, we only consider the following versions of mediator configuration: a local application (backed with hardware attestation), a remote trusted third party, or any party that uses confidential TEE.

	Mediator-Verifier	**Mediator-Prover**
Unlinkability	None: ✗* TEE: ✗* C-TEE: ✓	✓
Attribute Unforgeability	✓	None: ✗ TEE: ✓ C-TEE: ✓

TABLE 5.2: Unlinkability and attribute unforgeability properties for colluding parties of the FIDO-AC system.
* - considering ICAO eID, for other eID schemes a stronger unlinkability property can be achieved

The mediator (local or remote) colluding with either verifier or prover introduces a threat of breaking the properties of the FIDO-AC system. In Table 5.2, we evaluate privacy (i.e., unlinkability) and attribute unforgeability properties considering colluding mediator and various execution environments. The collaboration of the mediator and

verifier breaks the unlinkability property because data received by the verifier can be linked to the corresponding eID, unless a confidential TEE is used (i.e., user identifier not revealed to the mediator). For the mediator colluding with the prover, the attribute unforgeability property can be compromised if the TEE environment is not used (i.e., any proof can be generated). Therefore, we claim that a local verifier provides better privacy guarantees for privacy-oriented systems, whereas a remote mediator is more suitable for highly secure configurations.

The practical implementation of mediators relies on a secure trusted environment. Notably, we acknowledge the threats of jailbreaking TEE or leaking information from confidential TEE (e.g., side-channel attacks for SGX enclave[49; 159]). However, we argue that for the majority of use cases, we can safely assume that the TEE properties hold. Regarding trusted third party, we argue that, though it guarantees privacy and soundness of the system, it does not provide incentives for the running entity, and thus reduces the practical value. Therefore, considering the above threats, we decided to implement the FIDO-AC system (see Section 5.7) with a local mediator.

5.6.9 Web Security

The FIDO-AC system alters the execution of the application on the client side (i.e., web browser) by introducing a *fidoac.js* library. The security of loading and executing the *fidoac.js* script, similarly to any JavaScript library, depends on the deployment method (e.g., CDN or same origin) and applied web security mechanisms such as Sub-resource Integrity (i.e., verifying the script's hash) and Content Security Policy (i.e., restricting the script's origin). Considering the integration method of *fidoac.js* (i.e., a decorator pattern), we argue that our modification of the *navigator.credentials* object does not change the security properties of the parent application. Additionally, we consider the security of the communication between *fidoac.js* and the FIDO-AC client extension (e.g., FIDO-AC mobile application) depends on the platform-specific mechanisms, however, unintentional verification is unlikely because sharing the credentials involves user actions (e.g., providing an eID to verify liveliness). Nevertheless, if the channel is insecure, an adversary could send the request at the right time (i.e., during

a valid transaction) to launch a hijacking attack, hoping that the user does not notice any difference in the transaction data (similarly to MFA fatigue attacks). Therefore, we claim that the security of *fidoac.js* extension relies only on the web application configuration, browser security, and OS means to communicate with the FIDO-AC client extension.

5.7 Implementation and Evaluation

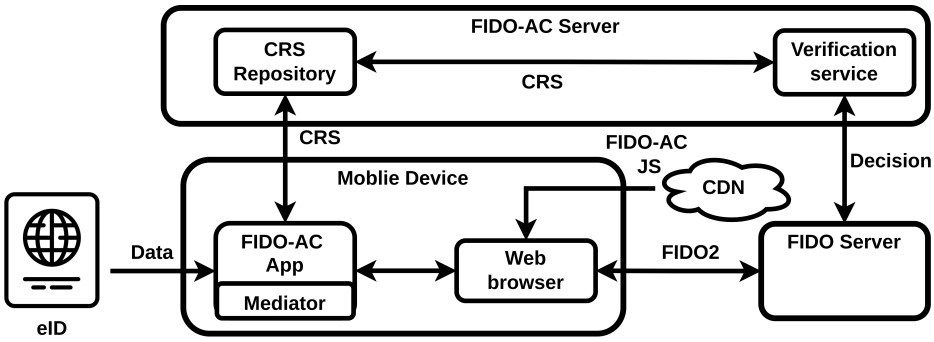


FIGURE 5.3: FIDO-AC system implementation high-level view.

In this section, we describe our approach to building the FIDO-AC system and demonstrate its feasibility in practical deployment in terms of performance evaluation. Notably, our prototype is one possible instantiation, and we can effortlessly adjust to any requirements. Following the system requirements (see Section 5.3), our implementation is suitable for most existing FIDO2 deployments. We achieved the claimed requirements by encapsulating and extracting the FIDO-AC-specific logic and then introducing the integration points. The high-level view of our FIDO-AC system implementation is presented in Figure 5.3. More details on the interaction between different components can be found in Appendix C.1. A detailed description of the system elements and the discussion about the design decisions can be found in Appendix C.2. The source code is published in our open-sourced repository (`https://github.com/FIDO-AC/fidoac`).

5.7.1 Implementation

The FIDO-AC system implementation is based on three components. The first is a user-centered mobile application for Android OS that introduces a bridge between the eID (in our prototype, an ICAO-based ePassport) and the FIDO authentication mechanism. We reuse an NFC interface to execute the PACE (or alternatively BAC) protocol and implement usability enchancements such as extracting data from the machine readable zone (MRZ) and data caching. We decided to follow a local mediator approach which is designed as a part of the FIDO-AC application. It relies on the security assumptions of Android's TEE. The zero-knowledge proof generation functionality follows the Groth'16 ZK-SNARK [92] using ported versions of rust-arkwork [54] and libsnark [125] libraries. The security as well as ZKP selection considerations can be found in Appendix C.2). The second component, FIDO-AC Server, is introduced to simplify the integration with existing FIDO implementations. It facilitates the anonymous credential trusted setup and verification. We recognized that the centralized and externalized party for the server-side process (e.g., ZKP verification), contributes to the usability and deployability of the FIDO-AC system. Similarly, the JavaScript integration (i.e., *fidoac.js*) is prepared to seamlessly introduce the FIDO-AC modifications to the browser execution. The script is served from CDN and automatically overrides the *navigator.credential* functions.

5.7.2 Performance Evaluation

We tested the FIDO-AC system using a Google Pixel 6 Pro and a Standard D4s v3 Microsoft Azure cloud instance (4 vcpus, 16 GiB memory). Without loss of generality, we use an ePassport as the ICAO-compliance eID. The performance of the running time is across 100 runs of the measured component. Reading the eID data takes about $\sim 1050ms$ in itself. Fortunately, this operation can be cached in the application's memory. The other part of the interaction with the eID (i.e., secure channel establishment via BAC/PACE and liveliness test with a local mediator) takes $\sim 740ms$. Verifying the ZKP is the least taxing operation, which only takes $< 10ms$ due to the usage of

ZK-SNARK. The downside of using ZK-SNARKs is the proving time. The proving time for our example (i.e., data and age policy proof) is $\sim 3.3s$ which is relatively slow compared to the verification time. Fortunately, the FIDO-AC application can precompute such proofs since they do not depend on any values chosen by third parties but only on data and randomness chosen by the application. The application can do it since the space of practical queries is small, or the user can predefine a set of accepted predicates. Assuming a completed offline preprocessed proof and cached eID data, the added latency of the FIDO-AC system compared to standard FIDO2 is less than $1s$ for our implementation.

Operation	**Platform**	**Time (ms)**	**SD (ms)**
eID Reading	Mobile	1059.4/0.0†	37.58
Liveliness Check	Mobile	738.92	47.06
ZK Verify	Cloud PC	8.19	0.29
ZK Prove	Mobile	3375.61*	95.25

† The FIDO-AC application can cache the read data.
* Running time reduces significantly with preprocessing.

TABLE 5.3: Performance overview of the various FIDO-AC operations. Time is averaged over 100 executions.

5.8 Discussion

Achieving high usability was the primary goal of our design. Therefore, we inspect the usability of FIDO-AC compared to other schemes. The user must install a separate FIDO-AC application to use the FIDO-AC system. Note that different relying parties' services can reuse the same application. Thus, the initial setup phase is one-time for all future FIDO-AC-enabled services. To reduce user friction and improve usability, users can opt into an optional document data caching feature to prevent inputting the document data in subsequent runs if they are comfortable with it. The FIDO-AC application can provide optical character recognition (OCR) functionality to read the

necessary information from the document's machine-readable zone (MRZ), eliminating the need for error-prone and high-friction manual input. It is worth mentioning that the usability of FIDO-AC (with cached eID data) is comparable to the FIDO2 authentication via the NFC channel (i.e., eID must be placed close to the reader).

In the case of the relying parties, we focused our efforts on the usability of deployment and integration. The relying party only has to include the provided JavaScript file into the web application page containing the call to the browser WebAuthn API without making any other changes to the existing web application codebase. To simplify and smoothen the integration of the verification logic needed by FIDO-AC into the existing FIDO infrastructure, we provide a docker image containing the verification service and its dependencies are provided. Alternatively, the relying party can choose to invoke the verification service hosted by the FIDO-AC server and parse the result according to the relevant business logic, which further minimizes the integration effort.

Electronic Identification Schemes. Digital transformation has been a strategic target for many countries, including issuing digital identity documents and remote identity verification. In particular, frameworks following the Issuer, Verifier, and Holder model, such as mobile driving license (mDL)[107] or eIDAS EUDI Wallet[52], are being introduced as a legal means to identify people. The FIDO-AC framework (excluding privacy advancements) resembles the abovementioned schemes with some key differences. FIDO-AC leverages existing eID documents, and thus it does not mandate to have a Holder role. In consequence, FIDO-AC does not introduce additional provisioning and enrollment procedures. Similarly, unlike generic eID frameworks, FIDO-AC proposes a specific set of technologies (e.g., ZKP and FIDO2) that, even though might not be suitable for every deployment, reduces the development and interoperability efforts. Using external eID, FIDO-AC implements roaming attributes (a concept similar to the FIDO2 roaming authenticator), enabling device-independent identity verification (e.g., through thin clients or kiosks). Regarding privacy, the mediator-based setup of FIDO-AC allows for a multi-show of credentials, which cannot be easily achieved with static credentials (e.g., a new set of credentials has to be issued to remain private, which is a noticeable inconvenience for both Issuer and Holder). Notably, FIDO-AC

and wallet-based schemes such as mDL are not exclusive. On the contrary, FIDO-AC can utilize those schemes if they provide an active authentication feature.

5.9 Conclusion

This chapter described and demonstrated an end-to-end solution for enforcing privacy in attribute-based authentication using the FIDO2 protocol. Our design considers usability for both users and implementers and is thus ready for production deployment. We integrate the eID environment and enforce a liveliness verification to increase the trustworthiness of the presented attributes, preventing the problem of attribute sharing. We leverage zero-knowledge proofs to guarantee the privacy of the attributes presentation. We introduce a custom FIDO2 extension for transporting anonymized credentials and present a mechanism to overcome the WebAuthn API limitations. We support our design with security and performance evaluations and a prototype implementation of the system components.

The FIDO-AC system, the core contribution of this thesis, provides a novel approach to the authentication and authorization processes, while simultaneously ensuring that the system is easy to integrate into existing IAM deployments. It incorporates our findings on the usability and privacy of the FIDO2 protocol and promotes the *"privacy by default"* principle. Importantly, it simplifies the integration of advanced privacy-enhancing technology into production systems through an industry-grade protocol. This design, supported by extensive research, provides clear guidance to organizations seeking to improve the privacy of their users' data.

6

Discussion, Future Works and Conclusion

6.1 Perception of Privacy in the IAM Industry

Integration of PETs technologies, such as the one presented in Chapter 5, is still not a priority for industries, including IAM systems. Privacy by design and privacy by default principles [44], although known for a long time, find marginal application. In Chapter 3, we discussed the challenges of introducing a privacy-preserving authentication method (i.e., FIDO2) from the organizational perspective; however, this is only one of the factors contributing to the limited use of PETs. Below, we discuss other factors that prevent the rapid integration of PETs into IAM solutions.

Identity and access management systems are generally operated in interconnected environments based on trust and consent. For example, federated identity management (FIM) allows sharing users' data between parties. However, the loosely coupled

mechanism of federated protocols (e.g., OAuth 2.0 or OpenID Connect) leaves room for oversharing data (e.g., requesting more PII than required). As demonstrated by Dimova et al. [60] in their large-scale evaluation of OAuth 2.0 parties, privacy principles and legislations (e.g., GDPR compliance) are notoriously broken. The above example shows that even a simple privacy mechanism (i.e., data minimization) is difficult to achieve. Therefore, the application of advanced PETs in the IAM environment, with thousands of parties, each having its own objectives, priorities, and budgets, poses significant engineering and standardization challenges.

From a software development perspective, the integration of privacy-enhancing technologies strongly relies on awareness, proper understanding, and engagement of software creators. Unfortunately, as presented by Tahaei et al.[177], privacy-related questions and answers on Stack Overflow, an open community of software developers, tend to focus on simple solutions such as anonymization or tokenization. More advanced methods, such as those used in the design of the FIDO-AC framework (e.g., zero-knowledge proofs), are rarely mentioned. Interestingly, the perspective of software developers on privacy varies depending on the use case. Notably, when integrating with 3rd party services, the question of data privacy arises; however, as indicated by Utz et al.[185] study, developers mostly draw their attention to analytics providers. Privacy concerns related to authentication and authorization (e.g., federated login) did not seem to be a prominent issue.

Privacy-enhancing technologies do not always align with business goals and current operations, particularly when it comes to data collection and analysis processes that rely on identity and access management for accurate data. These operations often show reluctance towards adopting privacy-enhancing technologies to enhance individuals' privacy. For instance, a user study conducted by Garrido et al. [81] among privacy practitioners in the industry sheds light on the reasons why advanced privacy-enhancing technologies, such as differential privacy, are not widely utilized. According to the authors, privacy is still perceived as a secondary priority compared to security, and is still not well understood (e.g., technological and know-how issues). Importantly,

respondents of the study pointed out that existing processes, analysis pipelines, and reporting heavily depend on personally identifiable information, making the introduction of PETs unfeasible.

Finally, from the end user perspective privacy is a complex and nuanced aspect which does not clearly translate to the technologies used in the industry. As presented by Colnago et al.[51], the behaviours of users seeking privacy are not always consistent. Within the context of IAM, the authors conducted an investigation pertaining to biometric identification, specifically facial recognition. The findings revealed that users exhibit hesitancy towards adopting privacy-impacting technologies, yet express a willingness to consider their utilization in the future. This discrepancy highlights the inconsistencies in users' decision-making processes concerning technology selection and the delicate balance between usability and privacy considerations. Regrettably, such inconsistency poses a significant obstacle to the rapid adoption of PETs within the industry, as it renders the prediction of users' preferences a challenging endeavor.

6.2 Privacy vs Accountability

The primary objective of privacy-enhancing technologies is to embody fundamental data protection principles, aiming to enhance individuals' anonymity by minimizing the use of personal data. This desirable feature effectively benefits users by addressing concerns related to consent tracking and PII data leakage. In contexts of honest and benign behavior, increased privacy provides advantages to both users, as their PII data is neither shared nor collected and companies, by reducing the risk associated with data storage. However, in the context of malicious actions, privacy can conflict with accountability. Contemporary identity solutions, such as decentralized identity and blockchain-based currencies such as ZCash, strive to strike the right balance without entirely sacrificing privacy. An interesting observation lies in the financial sector, where regulations such as "Know Your Customer" (KYC) and "Anti-Money Laundering" (AML) mandate accountability of identity and access management systems. These regulations require financial and banking institutions to be able to ascertain the

identities and values of parties involved in any transaction. This creates seemingly contradictory requirements between transaction privacy and user anonymity on one hand, and regulatory obligations like KYC/AML on the other, posing a significant obstacle to the widespread adoption of PETs.

In the realm of academia, the proposition of approaches that uphold both privacy and accountability has received attention. For example, Dauterman et al. [58] presented a system aimed at resolving privacy concerns associated with well-established industry authentication protocols (e.g., FIDO2 and TOTP). Furthermore, Maram et al. [131] addressed the interplay between accountability and privacy within the framework of decentralized identity. Despite these scholarly efforts, the aforementioned solutions remain at a considerable distance from achieving widespread implementation by industry stakeholders and regulatory entities.

6.3 Fragility of Privacy

Although often neglected, privacy is a highly desired and potentially life-saving property. For example, in scenarios involving whistleblowers or crown witnesses, maintaining privacy is of utmost importance. Interestingly, privacy has been perpetually a target of attacks. Accusations, such as the notion that only individuals with ill intentions seek refuge behind the privacy curtain, are commonplace among privacy skeptics. Conversely, supporters of privacy agree that everyone has the right to safeguard their personal lives from unwarranted intrusion. Irrespective of whether motivated by benevolent or malicious intentions, privacy is challenging to achieve and easily susceptible to compromise. When considering the case of crown witnesses, concealing all personal information proves to be a formidable task, and even parcially linkable information can lead to fatal consequences.

Technically, privacy objectives, as defined by Pfitzmann et al. [155], can be oriented toward properties such as anonymity, unlinkability, unobservability, and pseudonymity. Interestingly, these aforementioned properties are closely related to the context of execution. Therefore, even though theoretical privacy properties (e.g., unlinkability)

are proven, real-world implementations often suffer from contextual information leakage that undermines privacy guarantees. Our timing attack on FIDO2 authenticators demonstrates that even in well-established protocols implemented by leading companies, contextual information (in our case, processing time) can leak subtle information that compromises privacy. Our attack is not the only one to highlight the fragility of privacy. For instance, Tramèr et al. [183] presented a side-channel attack on Zcash, where the authors successfully deanonymized transactions based on time measurements.

6.4 Conclusion

Identity and access management systems serve as the forefront for interactions with digital systems. Whether used by humans or digital actors, these IAM systems are responsible for collecting a significant amount of data. This data collection occurs for various purposes, ranging from justified objectives such as correct processing (e.g., authorization decisions) to privacy-violating activities such as nonconsensual tracking. Unfortunately, IAM systems often tend to overcollect data, which is more and more noticeable as privacy awareness is slowly but steadily increasing in mainstream society. Consequently, the industry has begun shifting towards privacy-oriented solutions. Within the customer space, the significance of privacy in customer IAM has been acknowledged as an urgent issue, prompting the development of legislation, technical solutions, and standardization efforts.

In this thesis, we closely examined the privacy aspects of fundamental IAM processes, specifically authentication and authorization, in the context of the solutions employed in real-world scenarios. The research concentrates on the FIDO2 protocol, which stands as a state-of-the-art solution within the IAM industry. FIDO2, by design and default, prioritizes privacy preservation, and thus our research delves into the challenges encountered while also proposing innovative approaches to enchance FIDO2's privacy functionalities.

Throughout our research, we conducted an investigation into real-world challenges affecting the integration of FIDO2. Although FIDO2 has admirable support from the

community and major vendors, it faces early adaptation issues, such as the absence of know-how and implementation guidelines. Consequently, we shifted our focus to examining the extent of privacy guarantees in real-world deployments. During our study, we made a significant discovery - a remote side-channel attack exploiting the timings of authentication errors, which leaks information about authenticator identity. This finding not only contributes to the enhancement of FIDO2 authentication privacy but also exposes the fragility of existing privacy guarantees. Working closely with vendors, we successfully addressed this issue. With a solid understanding of the privacy mechanisms in FIDO2, we directed our efforts towards leveraging advanced PETs for IAM. In response, we developed a system called FIDO-AC, which combines FIDO2 with anonymous credentials. By incorporating industry-recognized protocols for zero-knowledge proofs, we demonstrated that our framework can be seamlessly integrated into existing deployments without compromising the formal security and privacy posture of FIDO2. Notably, our design yields a significant privacy improvement for both authentication and authorization processes.

This thesis demonstrates a comprehensive exploration of privacy in IAM solutions, complemented by a robust privacy-preserving system design and implementation. We are confident that our research will increase privacy awareness among industry professionals and attract the attention of academics to explore the practical applications of privacy-enhancing technologies in various industries.

A

Supplementary material for Chapter 3 - FIDO2 Integration Usability Challenges

A.1 FIDO2 Participants

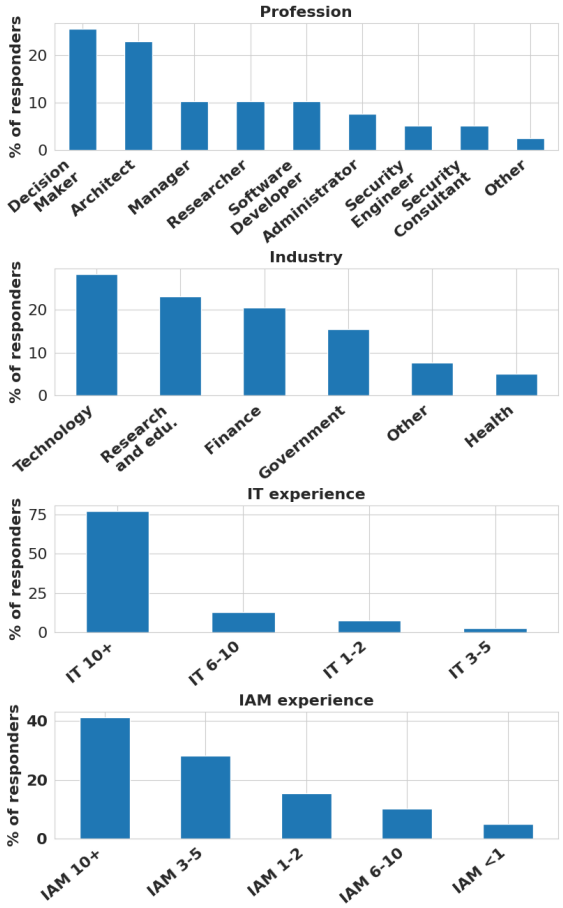


Figure A.1: Profile of respondents answering FIDO2 section.

A.2 Experience of All Participants

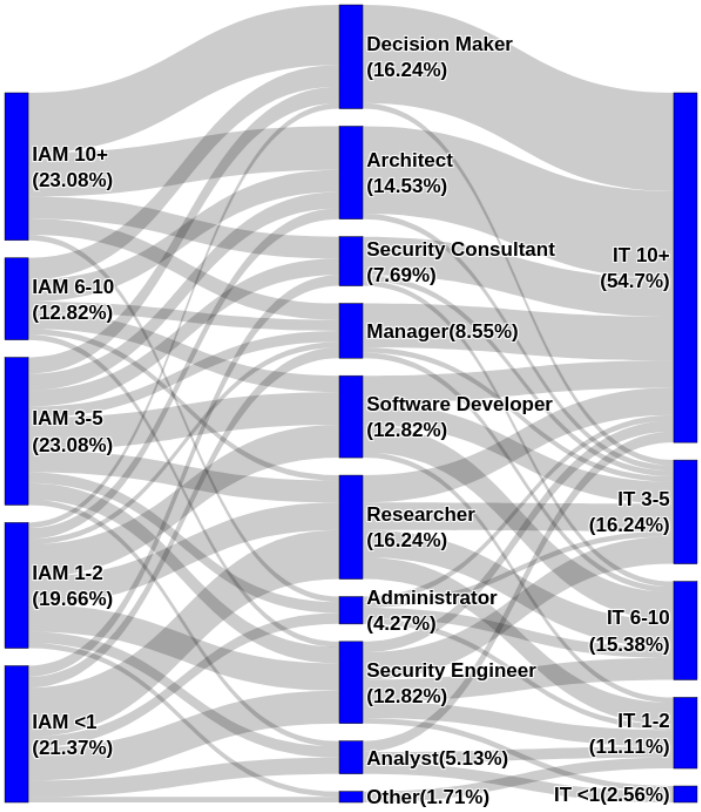


FIGURE A.2: Distribution of experience in Identity and Access Management (IAM) and Information Technology (IT) of respondents' professions

A.3 Kruskal-Wallis Relations

	Q1	Q2	p	V	effect
0	10	12	0.035	8.583	small
1	3	14	0.000	21.965	large
2	4	14	0.012	6.331	small
3	7	14	0.006	12.348	medium
4	13	14	0.000	26.086	large
5	15	14	0.000	46.035	large
6	18	14	0.022	5.227	medium

FIGURE A.3: Kruskal-Wallis test results ($p < 0.05$).

A.4 Cramér's V Relations

	Q1	Q2	Q1 val	Q2 val	p	V	effect	Confusion matrix			
								TP	FP	FN	TN
0	3	20	Less than one year	No challenges	0.000	0.625	large	0.400	0.600	0.000	1.000
1	8	22	I'm not sure	None	0.000	0.703	large	1.000	0.000	0.012	0.988
2	1	22	Software Developer	None	0.000	0.621	large	0.400	0.600	0.000	1.000
3	7	22	Yes (3-9 countries)	None	0.000	0.518	large	0.286	0.714	0.000	1.000
4	6	23	250 - 999	None	0.000	0.606	large	0.400	0.600	0.000	1.000
5	2	17	3 - 5 years	I'm not sure	0.000	0.562	large	1.000	0.000	0.053	0.947
6	3	24	6 - 10 years	Not sure	0.000	0.322	medium	0.111	0.889	0.000	1.000
7	3	18	More than 10	Yes	0.001	0.552	large	0.938	0.062	0.391	0.609
8	2	15	More than 10	Yes	0.001	0.316	medium	0.469	0.531	0.170	0.830
9	3	23	1 - 2 years	None	0.001	0.545	large	0.333	0.667	0.000	1.000
10	1	24	Manager	Not sure	0.001	0.304	medium	0.100	0.900	0.000	1.000
11	3	15	More than 10	Yes	0.001	0.301	medium	0.593	0.407	0.256	0.744
12	3	17	Less than one year	SaaS	0.002	0.497	medium	1.000	0.000	0.135	0.865
13	2	23	1 - 2 years	Hardware roaming	0.002	0.492	medium	1.000	0.000	0.194	0.806
14	8	23	I'm not sure	I don't know	0.003	0.480	medium	1.000	0.000	0.079	0.921
15	7	23	Yes (2 countries)	I don't know	0.003	0.480	medium	1.000	0.000	0.079	0.921
16	4	18	Yes	Yes	0.003	0.474	medium	0.818	0.182	0.353	0.647
17	1	18	Decision maker	Yes	0.004	0.464	medium	1.000	0.000	0.483	0.517
18	10	23	No	None	0.004	0.458	medium	0.250	0.750	0.000	1.000
19	5	17	Technology	SaaS	0.005	0.449	medium	0.455	0.545	0.071	0.929
20	3	18	3 - 5 years	Yes	0.006	0.441	medium	0.273	0.727	0.750	0.250
21	1	23	Architect	None	0.008	0.424	medium	0.222	0.778	0.000	1.000
22	10	23	No	Platform	0.008	0.424	medium	0.500	0.500	0.097	0.903
23	9	17	Only cloud	SaaS	0.009	0.420	medium	0.600	0.400	0.118	0.882
24	3	17	1 - 2 years	I'm not sure	0.010	0.410	medium	0.333	0.667	0.030	0.970
25	1	23	Manager	Mobile app	0.011	0.405	medium	1.000	0.000	0.343	0.657
26	6	17	250 - 999	Both	0.014	0.393	medium	0.000	1.000	0.588	0.412
27	3	18	1 - 2 years	Yes	0.014	0.393	medium	0.167	0.833	0.697	0.303
28	9	23	Only cloud	I don't know	0.019	0.376	medium	0.400	0.600	0.059	0.941
29	5	16	Health	I'm not sure	0.021	0.369	medium	0.500	0.500	0.054	0.946
30	1	17	Security Engineer	I'm not sure	0.021	0.369	medium	0.500	0.500	0.054	0.946
31	7	18	No (only 1 country)	Yes	0.022	0.368	medium	0.412	0.588	0.773	0.227
32	9	17	Yes - both	SaaS	0.026	0.356	medium	0.121	0.879	0.500	0.500
33	5	23	Technology	I don't know	0.028	0.352	medium	0.273	0.727	0.036	0.964
34	9	23	Only on-prem	Platform	0.030	0.347	medium	1.000	0.000	0.158	0.842
35	7	17	Yes (2 countries)	SaaS	0.030	0.347	medium	1.000	0.000	0.158	0.842
36	7	16	Yes (10 or more)	I'm not sure	0.031	0.346	medium	0.188	0.812	0.000	1.000
37	6	23	below 50	Hardware roaming	0.035	0.337	medium	0.571	0.429	0.188	0.812
38	6	17	250 - 999	Software	0.036	0.336	medium	0.600	0.400	0.176	0.824
39	4	17	Yes	I'm not sure	0.040	0.328	medium	0.000	1.000	0.176	0.824
40	10	17	No	Software	0.043	0.325	medium	0.500	0.500	0.161	0.839
41	9	23	Yes - both	I don't know	0.043	0.324	medium	0.061	0.939	0.333	0.667
42	5	18	Research & edu.	Yes	0.047	0.318	medium	0.333	0.667	0.700	0.300
43	10	18	I am not certain	Yes	0.048	0.317	medium	0.286	0.714	0.688	0.312

Table A.1: Cramér's V (χ_2) test results for $p < 0.05$ and medium and large effects.

B

Supplementary material for Chapter 4 - FIDO2 Privacy -Timing Attacks

B.1 Responsible Disclosure

The results of our research were sent to the responsible parties. We notified three vendors of web browsers (Chromium, Firefox, and Safari), two hardware tokens vendors (Feitian and Hypersecu) and the FIDO Alliance.

For Chromium, we sent the vulnerability report though the chromium bug tracing platform. After a brief explanation, the chromium security team acknowledged the vulnerability. On 14th September 2021, we received a plan how the mitigation will be implemented:

"I think the mitigation step we should take is to deduplicate allowedCredentials before

making CTAP requests. This should eliminate the amplification, and then the time it takes the user to touch the key would hopefully dominate any difference in CTAP request time. We can also limit the size of that list explicitly, but we would need to be careful not to break any weird outlier sites with users that have lots of enrolled credentials. As for severity, per https://www.chromium.org/developers/severity-guidelines I would say the cross-origin user correlation is a type of information disclosure." - martinkr@google.com

On 5th October 2021 the first implementation which included a deduplication mechanism combined with a limitation of the size of the allowCredential list (limit set to 64 key handles) was provided. We tested the vulnerability fix on the canary release of Chrome browser and confirmed that the attack is mitigated. The vulnerability received id CVE-2021-38022.

Similarly, Firefox was notified through their bug tracing channel. The Firefox team recognized the issue and assigned internal bug id 1730434. On 15th September 2021, the issue status was changed from unconfirmed to new. At the time of writing, the implementation of the mitigation mechanism is in progress.

In case of the Safari browser, we sent an email to the product security team. On 28th September 2021, we received a confirmation saying:

"We don't automatically provide status updates on issues as we work on them. We will reach out if we have any questions or need additional details.".

Our report received tracking number 781222840 and is undergoing an investigation.

We reached out to Feitian through their official contact email. On 14th September, our report was passed to Feitian's internal team:

"I forward your report to R&D team. Our developers are investigating it, we will update you when we came to conclusion." - lena@ftsafe.com

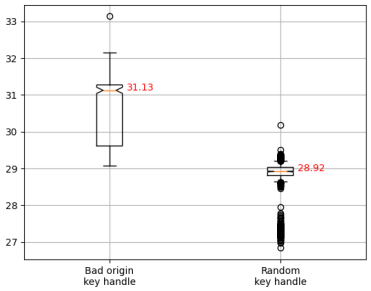
We explained the issue to the Feitian's engineering team and provided the detailed recording of the testing procedure. The issue is being investigated.

The report to Hypersecu was sent to their official contact email. We received their acknowledgment on 13th September 2021. We provided an exhaustive description of the problem together with results and recording of our tests. On 15th October 2021

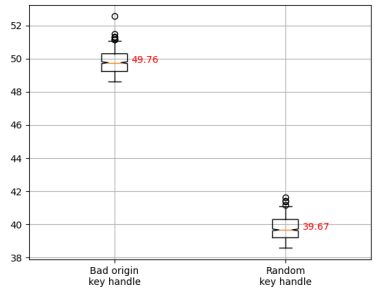
we received a request for additional parameters describing the vulnerable token:
"Could you please use the following tool to send the HyperFIDO info to us? We want to make sure the version info of the key you tested. On the other hand, would you mind letting us know from where you bought the key? Was it Amazon AU?" - james@hypersecu.com
We provided the requested data. The Hypersecu team is investigating the problem.

Additionally, we sent our report to the FIDO Alliance Security Certification Secretariat. The Fido Alliance representative analysed our report and endorsed our findings:
"After a quick analysis, I'd like to congratulate you for your successful timing attacks conducted on multiple certified silent authenticators at L1." - roland@fidoalliance.org
Moreover, we learned that our contribution will help to spread awareness of the advantages of the highest certification levels (L3/L3+), which require deep laboratory analysis including timing measurements:
"Indeed, L1 certification is not designed to provide assurance against such attacks so, there isn't much here in terms of actions we could do in addition to notifying the vendors. However, I think that RPs should be made aware of such type of attack so they can better understand why L3/L3+ certification makes sense and probably help in creating incentives in this sense. And that's something where FIDO could potentially help based on your findings." - roland@fidoalliance.org

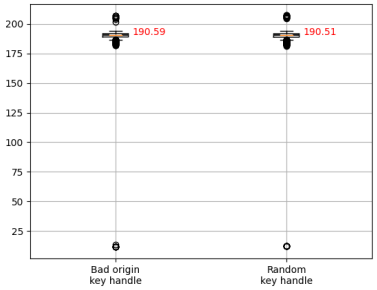
B.2 Silent Authentication Measurements



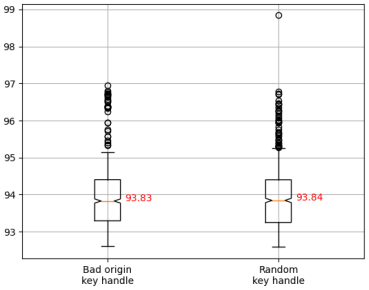
(a) Silent authentication times for Feitian K26



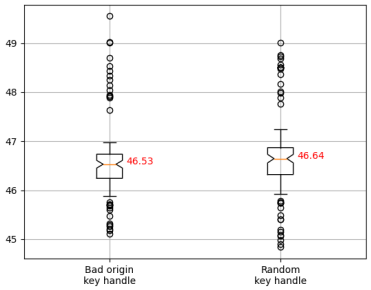
(b) Silent authentication times for HyperFIDO Titan Pro



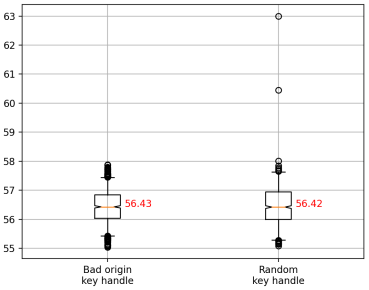
(c) Silent authentication times for Yubikey 5 (samples below 20ms represent initial calls without triggering the defence mechanism)



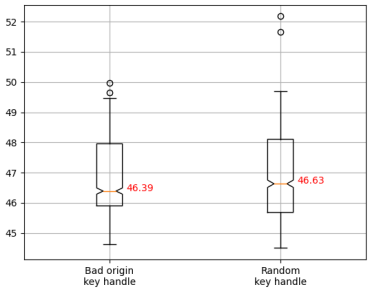
(d) Silent authentication times for Token2 T2F2 Bio



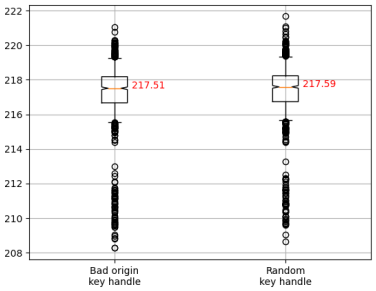
(e) Silent authentication times for TrustKey G320H



(f) Silent authentication times for Google Titan



(g) Silent authentication times for VeriMark Guard Fingerprint



(h) Silent authentication times for AuthenTrend ATKey.Pro

Figure B.1: Silent authentication time[ms] measurements.

B.3 Attack Scenarios

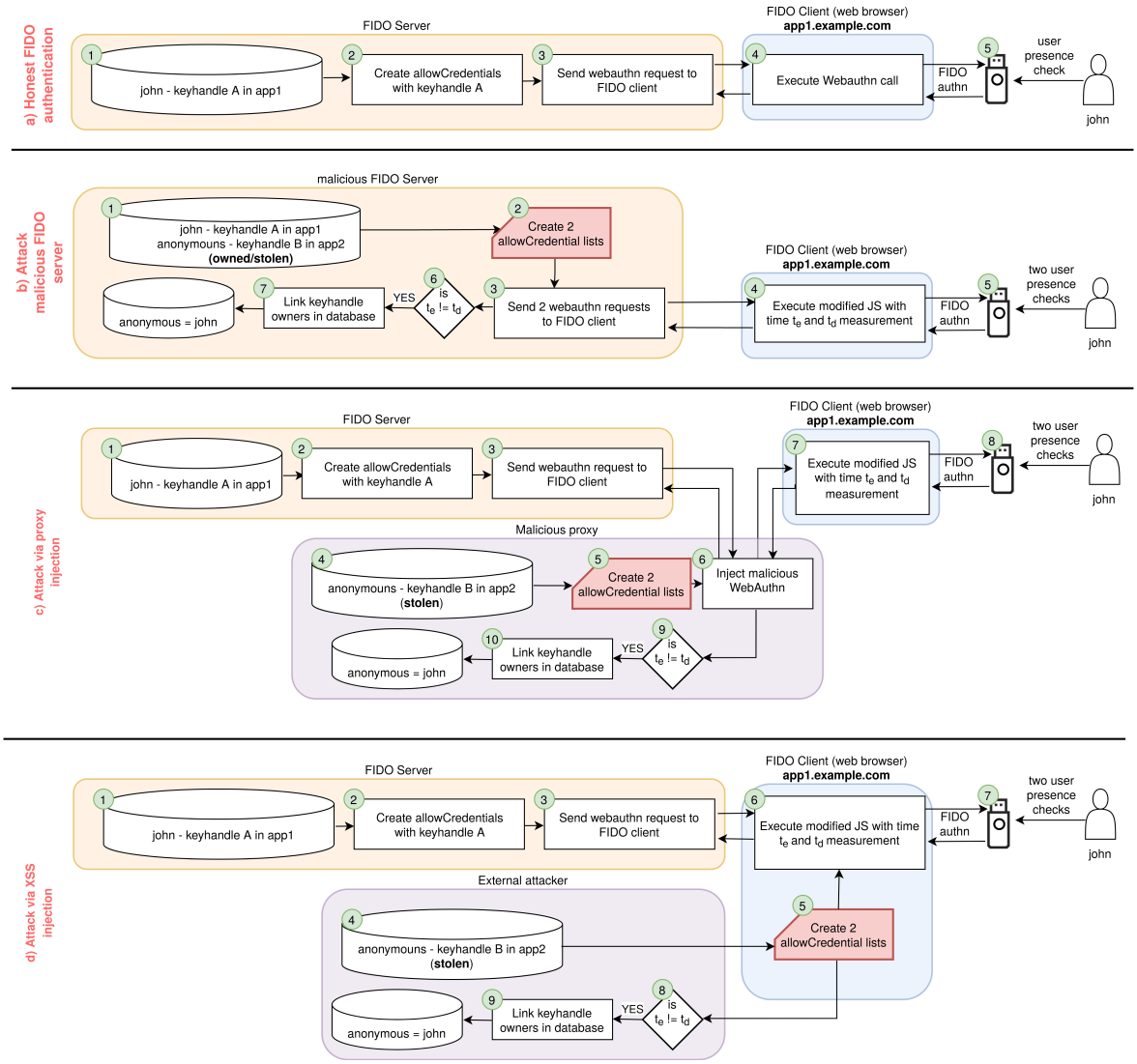


Figure B.2: FIDO timing attack scenarios.

a) An honest FIDO2 authentication. The FIDO Server uses John's keyhandle (A) to trigger an WebAuthn call in the FIDO Client.

b) A malicious FIDO Server, which is in possession of John's keyhandle as well as a keyhandle of an anonymous user. The FIDO server executes the attack to learn if the anonymous user is in fact John. In step 2, two *allowCredential* lists are generated and sent to the FIDO client to execute and measure two consecutive WebAuthn calls (step 4). Then the FIDO server decides if times differ (step 6) and if they do, the identity is linked (step 7).

c) An attack by a malicious proxy. In this configuration, the FIDO Server is honest, however the proxy manipulates messages and JavaScript calls to learn the timing difference (step 6). The Javascript execution and linking decision is the same as scenario (b).

d) Attack via injection of malicious JavaScript. In this case, an adversary manipulates WebAuthn calls directly from JavaScript (steps 5 and 6). The decision and linking process remain unchanged.

B.4 Open Source Implementations of Key Handling

```
259     // Decrypts a credential ID and writes the private key into a
260     // PublicKeyCredentialSource. None is returned if the HMAC test fails
261     // or the relying party does not match the decrypted relying party ID hash.
262     pub fn decrypt_credential_source(
263         &self,
264         credential_id: Vec<u8>,
265         rp_id_hash: &[u8],
266     ) -> Result<Option<PublicKeyCredentialSource>, Ctap2StatusCode> {
267         if credential_id.len() != CREDENTIAL_ID_SIZE {
268             return Ok(None);
269         }
270         let master_keys = self.persistent_store.master_keys()?;
271         let payload_size = credential_id.len() - 32;
272         if !verify_hmac_256::<Sha256>(
273             &master_keys.hmac,
274             &credential_id[..payload_size],
275             array_ref![credential_id, payload_size, 32],
276         ) {
277             return Ok(None);
278         }
279         let aes_enc_key = crypto::aes256::EncryptionKey::new(&master_keys.encryption);
280         let aes_dec_key = crypto::aes256::DecryptionKey::new(&aes_enc_key);
281         let mut iv = [0; 16];
282         iv.copy_from_slice(&credential_id[..16]);
283         let mut blocks = [[0u8; 16]; 4];
284         for i in 0..4 {
285             blocks[i].copy_from_slice(&credential_id[16 * (i + 1)..16 * (i + 2)]);
286         }
287         cbc_decrypt(&aes_dec_key, iv, &mut blocks);
288         let mut decrypted_sk = [0; 32];
289         let mut decrypted_rp_id_hash = [0; 32];
290         decrypted_sk[..16].clone_from_slice(&blocks[0]);
291         decrypted_sk[16..].clone_from_slice(&blocks[1]);
292         decrypted_rp_id_hash[..16].clone_from_slice(&blocks[2]);
293         decrypted_rp_id_hash[16..].clone_from_slice(&blocks[3]);
294         if rp_id_hash != decrypted_rp_id_hash {
295             return Ok(None);
296         }
```

Figure B.3: Key Decryption Function in OpenSK Token Implementation [88].

```
264 void generate_private_key(uint8_t * data, int len, uint8_t * data2, int len2,
265                                                     uint8_t * privkey)
266 {
267     crypto_sha256_hmac_init(CRYPTO_MASTER_KEY, 0, privkey);
268     crypto_sha256_update(data, len);
269     crypto_sha256_update(data2, len2);
270     crypto_sha256_update(master_secret, 32);     // TODO AES
271     crypto_sha256_hmac_final(CRYPTO_MASTER_KEY, 0, privkey);
272
273     crypto_aes256_init(master_secret + 32, NULL);
274     crypto_aes256_encrypt(privkey, 32);
275 }
```

Figure B.4: Pseudorandom Key Generation in SoloKeys Firmware Implementation [172].

C

Supplementary material for Chapter 5 - Fast IDentity Online with Anonymous Credentials (FIDO-AC)

C.1 FIDO-AC Implementation Interactions

Below, we describe the high-level interactions between the parties of the FIDO-AC system implementation. The illustration of the flow is presented in Figure C.1. The interaction with the FIDO-AC system starts when the user bootstraps the FIDO-AC application (step 1.). The process involves reading the data and signature from the eID document and caching them in the FIDO-AC application. After the initialization,

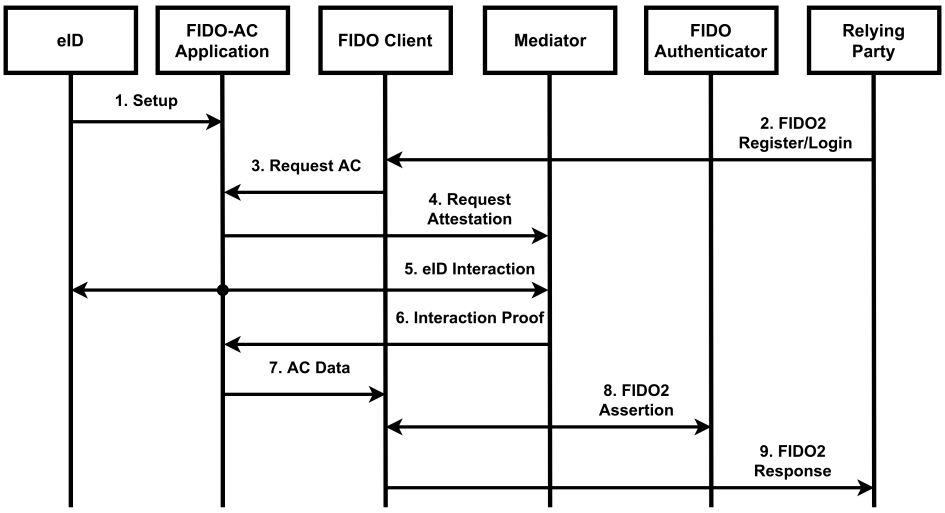


FIGURE C.1: FIDO-AC interaction flow.

the user can trigger the FIDO2 transaction (step 2.). The FIDO2 assertion request is intercepted in the FIDO client and passed to the FIDO-AC application (step 3.). At this stage, the FIDO-AC application starts interacting with the mediator (step 4.). The application shares the eID data: the hashed data values, eID public key, and signature, as well as the FIDO challenge and nonce. Notably, the mediator does not learn the identity of the user. However, because of the eID public key, the interaction of the same eID with the mediator is linkable. The mediator's task is to verify the authenticity of the data and prove the liveliness of the eID. To accomplish that, the mediator starts the interaction with the eID (step 5.). The details of the interaction are described in Section 5.5.2. The mediator returns a signature attesting to the liveliness (step 6.). It is worth noting that the relying party cannot link the mediator's signature to any particular eID because the signed message depends on its challenge and a hash value that is randomized using a 128-bit nonce. The FIDO-AC application generates a proof, which proves the knowledge of the hash preimage (i.e., DG data) and that this personal data is according to the policy Policy_S (e.g., above 18 years old). Both (att_m, σ_m) and π_{zkp} are sent back to the FIDO client (step 7.) and attached to the FIDO2 transaction (step 8.). Finally, the FIDO client sends the complete assertion to the relying party, which runs the signature and ZKP verification.

C.2 FIDO-AC Implementation Elements

FIDO-AC Application with Mediator: The application communicates with ePassport through an NFC interface with active NFC scanning triggered by user interaction. This approach enables the mobile application to avoid spawning unnecessary FIDO-AC processes on accidental proximity triggers of NFC events. The ePassport employs a password-based key-established protocol to protect the communication between the eID and the terminal. Therefore, our FIDO-AC application implements the PACE protocol to establish this secure channel. We also implemented the alternative BAC protocol that served as the backup for backward compatibility if the eID does not support PACE. PACE and BAC require a password as input, composed of user data such as document number, expiry date, and date of birth. Manual entry of the data is tedious and error-prone. The mobile application provides an optional optical character recognition (OCR) to scan, read and parse the necessary information on the document's machine readable zone (MRZ). Furthermore, the read data can be cached if the user opts in. Caching the personal data reduces the eID read time, which initially takes $\sim 1060ms$. Moreover, the cached data can be encrypted at rest to provide further protection (defense-in-depth), albeit at a slightly higher computational cost.

The mobile application also displays information about the origin of the FIDO-AC requester and its queries. It allows users to check whether they are expecting the origin and the associated queries. Before the eID is scanned, the user retains the right to cancel the transaction and downgrade to FIDO.

Once the reading of the eID is done, the FIDO-AC mobile application will communicate with a mediator as described in Section 5.5.2. In our prototype implementation, we opted for a local mediator implemented as part of the mobile application. This approach relies on the property that (for a secure system) hardware-backed keys can only be used by the application that generated it, and the TEE enforces such a boundary. To attest to an honest computation, the FIDO-AC mediator will use a package-bound hardware-attested key to sign the result described in Section 5.5.1. The relying party

verifies the signature and is assured about the honest behavior of the mediator component.

The zero-knowledge proof system selected to realize the privacy-preserving functionality is Groth'16 ZK-SNARK [92], mainly for its efficiency, short proof size, and the availability of existing implementation, rust-arkwork [54] and libsnark [125]. In this case, we chose to use rust-arkwork [54]. One of the additional deciding factors in choosing this library was the ease of cross compilation between ARM and x86. In FIDO-AC, we prove statements that use such examples as building blocks. Unfortunately, Groth'16 ZK-SNARK requires a trusted setup to generate a common reference string (CRS) for proving and verifying. The FIDO-AC server will be trusted to host an honestly generated CRS repository. We do not introduce new trust assumptions here since we already trust the FIDO-AC server to provide the certified FIDO-AC mobile application needed to secure the user-side mediator implementation.

During verification, the verification service checks the correct FIDO-AC response and whether the mediator's public key has a valid hardware-backed key attestation. A valid hardware-backed key attestation for this scenario requires the presence of the mediator package name, the mediator package's certificate fingerprint, and an attestation challenge that is the same as the FIDO challenge. Using this challenge, we bound the mediator's hardware-backed key to the particular FIDO session, which is supported thanks to the functionality provided by the Android API. It is possible to have a more robust integrity check utilized by PlayIntegrity API [89]. However, this particular approach is not considered for implementation because of the reliance on the GooglePlay service that might not be available for some Android devices.

FIDO-AC Server: One of our main goals in designing the FIDO-AC systems is to reduce the complexity of the deployment process. To meet this goal, we particularly focused on the design of the FIDO-AC Server. We prepared the Server as an independent element of architecture (i.e., implemented as a self-contained and stateless docker container). Notably, the Server is technology agnostic as it publishes a REST interface over HTTP. The main tasks of the Server are the following: distribution of the common reference string for the zero-knowledge proof system, verification of the mediator's

attestation, and the zero-knowledge proof created by the FIDO-AC application.

The ZKP trusted setup parameters generated by the system must be propagated to all parties using the proof system (i.e., prover and verifier). A naive method would be to include the parameters in the applications (e.g., in the configuration files). Even though this could work well with our FIDO-AC mobile application, integration on the verifier side could raise usability issues as it requires integration with a relying party (usually out of the FIDO-AC system control). Considering the learned lessons from the configuration problems of other complex security protocols (e.g., the TLS configuration issues reported by Krombholz et al. [120]), we decided to externalize and automate the parameter configuration process. Therefore, we modeled the FIDO-AC Server as a centralized repository for CRS data, which can be conveniently discovered using a single HTTP call.

The optional functionalities of the FIDO-AC Server are introduced to minimize the integration efforts. Notably, the proof verification functionality can be implemented as a local module (i.e., in the relying party). However, this approach does not scale well, as the great diversity of technologies used for commercial applications makes a single implementation of the verifier impossible. Therefore, we decided to encapsulate and extract this functionality to a separate component (i.e., FIDO-AC Server) which can be either local for the relying party (e.g., deployed next to the application) or hosted by an external trusted party. Similarly to the trusted setup functionality, the verifier can be reached by sending a simple HTTP request. Following our approach, the relying party can integrate with only marginal changes to its source code (i.e., one HTTP call).

FIDO2 Integration: As discussed in Section 5.5.3, implementing a fully functional FIDO2 extension is significantly limited. Therefore, the processing of the *fidoac* extension is implemented before and after WebAuthn API calls. We use the relying parties' challenge to bind the extension data to the FIDO assertion. Below, we briefly describe the FIDO2 flow enhanced with *fidoac* extension. The steps of the process are depicted in Figure C.2.

The flow starts with generating a FIDO assertion request with a *fidoac* extension

(step 1.). The processing of the *fidoac* extension is encapsulated inside *fidoac.js*, and thus it does not require any change of the existing FIDO JavaScript code. To achieve a frictionless integration, *fidoac.js* overwrites the original *navigator.credentials.get* function with a custom implementation (steps 2. and 3.). In consequence, *fidoac.js* can preprocess FIDO assertion and forward it to WebAuthn API (i.e., the original *navigator.credentials.get* function). The processing of the *fidoac* extension involves communication with the FIDO-AC service (steps 4. and 5.) using internal (i.e., localhost) HTTP calls. The binding between FIDO-AC data and FIDO transaction is done via appending the SHA-256 hash of data to the FIDO assertion challenge (step 6.). The modified request is passed to WebAuthn API (steps 7-10) to generate a signed assertion. Notably, our *fidoac* extension is not forwarded to the FIDO authenticator due to the web browser limitations, and thus the only signed modification is the appended part of the challenge. The response from WebAuthn API is again intercepted by *fidoac.js* and enriched with fidoac data (inside the *clientExtensionResults* element). The final response is sent back to the server (steps 11. and 12.). Because the challenge was modified, the FIDO server has to execute the same action (i.e., hashing the extension data and appending it to the challenge) to verify the FIDO2 transaction successfully.

C.3 FIDO2 Challenge

The FIDO2 protocol uses challenges to prevent replay attacks. In the FIDO-AC system, we use this mechanism as a binding mechanism (see Section 5.5.3). Even though theoretically (i.e., in the WebAuthn specification), the length of the challenge is not limited, the software implementations might trigger errors if the challenge is above a certain threshold. We studied the documentation of various FIDO2 authenticators, and we could not find any explicit limitations, which suggests that vendors do not artificially limit the challenge size and, thus, are compliant with the vast majority of the FIDO Servers.

We empirically tested various authenticators to ensure their compatibility with the

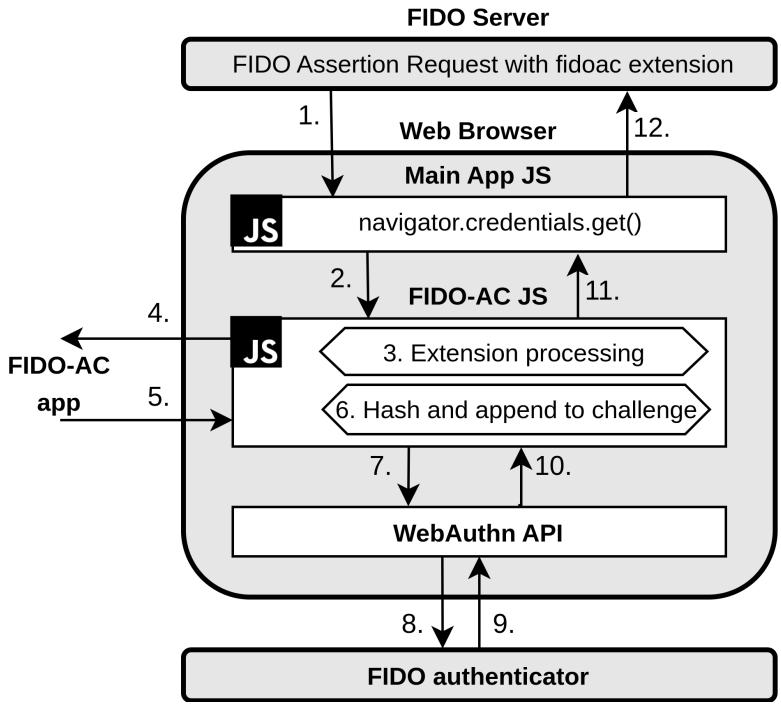


Figure C.2: FIDO-AC extension processing.

FIDO-AC system. In our test, we examined both platform and roaming authenticators using the Chrome browser on mobile. We tested Android and iOS platform authenticators and Yubico 5 roaming authenticators. The test procedure repeatedly triggers FIDO2 authentication, increasing the challenge size before each run. The evaluation stops when the threshold of 100Kb for the HTTP message is reached. We adopted this threshold from one of the HTTP servers (i.e., expressjs). The results confirm that authenticators allow significantly longer challenges. Therefore, we claim that the majority of commercial authenticators should support our approach of appending the hash value (256-bits of SHA-256).

C.4 FIDO Extension Considerations

The WebAuthn API implementations are known not to support custom extensions, and thus various solutions have been proposed. We analyzed academic and industry approaches and identified three ways to mitigate this problem. The intuitive one is bypassing client limitations by implementing a custom FIDO client and authenticator.

For example, Gou et al. [93] follow this approach to introduce a QR-based registration flow. Unfortunately, in our case, replacing popular FIDO clients and authenticators is not possible because of requirement **R.5** and requirement **R.6**. Okawa et al. [146] presented a solution closer to our needs, which used relying party code to implement WebAuthn API. It is a smart way to evaluate custom FIDO2 solutions. However, it is not suitable for production deployment. A solution that could be used in the wild is the one proposed by Putz et al. [158]. The authors solve the extension issue by implementing a plugin for web browsers (FIDO clients) that passes the extension content to the authenticator. Even though this approach solves the limited FIDO clients, it does not address requirement **R.5**. For example, in the mobile use case, browsers limit or forbid extensions, which makes integration difficult. Additionally, the arbitrary plugin raises adaptation and scalability challenges (requirements **R.5** and **R.6**) as it needs to be introduced for each user separately.

References

[1] 2023 Data Breach Investigations Report. Technical report, Verizon. URL: https://www.verizon.com/business/resources/T75/reports/2023-data-breach-investigations-report-dbir.pdf. 2, 12

[2] Hyperledger Indy. Technical report, Hyperledger Foundation. URL: https://www.hyperledger.org/use/hyperledger-indy. 28

[3] Android Now FIDO2 Certified, Accelerating Global Migration Beyond Passwords, Feb. 2019. URL: https://fidoalliance.org/android-now-fido2-certified-accelerating-global-migration-beyond-passwords/. 65

[4] Client to authenticator protocol (ctap), Jan. 2019. URL: https://fidoalliance.org/specs/fido-v2.0-ps-20190130/fido-client-to-authenticator-protocol-v2.0-ps-20190130.html. 65

[5] Microsoft achieves fido2 certification for windows hello, May 2019. URL: https://fidoalliance.org/microsoft-achieves-fido2-certification-for-windows-hello/. 65

[6] U.s. general services administration's rollout of fido2 on login.gov, Mar. 2019. URL: https://fidoalliance.org/u-s-general-services-administrations-rollout-of-fido2-on-login-gov/. 65

[7] Visa case study, Jan. 2019. URL: https://fidoalliance.org/visa-case-study/. 65

[8] Expanded support for fido authentication in ios and macos, July 2020. URL: https://fidoalliance.org/expanded-support-for-fido-authentication-in-ios-and-macos/. 65

[9] *Hacking MFA in General*, chapter 5, pages 122–140. John Wiley Sons, Ltd, 2020. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119672357.ch5, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119672357.ch5, doi:https://doi.org/10.1002/9781119672357.ch5. 12

[10] Google announcement, Oct. 2021. URL: https://blog.google/technology/safety-security/delivering-10000-security-keys-high-risk-users/. 66

[11] Media capture and streams, Mar. 2021. URL: https://www.w3.org/TR/mediacapture-streams/. 86

[12] National health service uses fido authentication for enhanced login, Feb. 2021. URL: https://fidoalliance.org/national-health-service-uses-fido-authentication-for-enhanced-login/. 65

[13] Recommended life spans to guide pc, mobile and other device replacement strategies, March 2021. URL: https://www.gartner.com/en/documents/4000060. 43

[14] A side journey to titan side-channel attack on the google titan security key, Jan. 2021. URL: https://ninjalab.io/wp-content/uploads/2021/01/a_side_journey_to_titan.pdf. 21, 64

[15] Web authentication: An api for accessing public key credentials, Feb. 2021. URL: https://www.w3.org/TR/webauthn-2/. 65

[16] Yubico announcement, Mar. 2021. URL: https://www.yubico.com/blog/yubico-donates-25000-yubikeys-to-microsoft-accountguard-customers-in-31-countries/. 66

[17] Digital identity hype cycle 2022, July 2022. URL: https://www.gartner.com/interactive/hc/4016895?ref=explorehc. 35

[18] From cookie theft to bec: Attackers use aitm phishing sites as entry point to further financial fraud, July 2022. URL: https://www.microsoft.com/en-us/security/blog/2022/07/12/from-cookie-theft-to-bec-attackers-use-aitm-phishing-sites-as-entry-point-to-further-financial-fraud/. 3, 12

[19] Owasp top 10, Feb. 2022. URL: https://owasp.org/Top10/. 78

[20] AARC. Guidelines for credential delegation, 2017. URL: https://aarc-project.eu/wp-content/uploads/2017/03/AARC-JRA1.4D.pdf. 42

[21] Y. Ackermann. WebAuthn and Passkey Awesome. https://github.com/herrjemand/awesome-webauthn. [Online; accessed 7-Sept-2023]. 46

[22] C. Adams. *Dictionary Attack (II)*, pages 147–147. Springer US, Boston, MA, 2005. doi:10.1007/0-387-23483-7_107. 11

[23] A. Alam, K. Krombholz, and S. Bugiel. Poster: Let history not repeat itself (this time) – tackling webauthn developer issues early on. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2669–2671, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3319535.3363283. 46

[24] R. AlHusain and A. Alkhalifah. Evaluating fallback authentication research: A systematic literature review. *Computers and Security*, 111:102487, 2021. URL: https://www.sciencedirect.com/science/article/pii/S0167404821003114, doi:https://doi.org/10.1016/j.cose.2021.102487. 44

[25] F. Alliance. FIDO security reference, 2021. URL: https://fidoalliance.org/specs/common-specs/fido-security-ref-v2.1-rd-20210525.html. 100

[26] O. G. Andrew Cameron. An Overview of the Digital Identity Lifecycle (v2) . https://bok.idpro.org/article/id/31/. [Online; accessed 7-Sept-2023]. 37

[27] ARM. TEE Reference Documentation. https://www.arm.com/technologies/trustzone-for-cortex-a/tee-reference-documentation. [Online; accessed 7-Sept-2023]. 43

[28] S. S. Arora, S. Badrinarayanan, S. Raghuraman, M. Shirvanian, K. Wagner, and G. Watson. Avoiding lock outs: Proactive fido account recovery using manager-less group signatures. Cryptology ePrint Archive, Paper 2022/1555, 2022. URL: https://eprint.iacr.org/2022/1555. 45

[29] AUSTRAC. Optus Data Breach. https://www.austrac.gov.au/optus-data-breach-working-our-reporting-entities, 2022. [Online; accessed 7-Sept-2023]. 2, 95

[30] Australian Cyber Security Magazine. Latitude Financial Scrambles to Contain Large Data Breach. https://australiancybersecuritymagazine.com.au/latitude-financial-scrambles-to-contain-large-data-breach/, 2022. [Online; accessed 7-Sept-2023]. 2

[31] M. Barbosa, A. Boldyreva, S. Chen, and B. Warinschi. Provable security analysis of fido2. In T. Malkin and C. Peikert, editors, *Advances in Cryptology – CRYPTO 2021*, pages 125–156, Cham, 2021. Springer International Publishing. 18, 35, 37

[32] M. Barbosa, A. Boldyreva, S. Chen, and B. Warinschi. Provable security analysis of FIDO2. pages 125–156, 2021. doi:10.1007/978-3-030-84252-9_5. 101, 102

[33] C. Baum, J. H. yu Chiang, B. David, and T. K. Frederiksen. Sok: Privacy-enhancing technologies in finance. Cryptology ePrint Archive, Paper 2023/122, 2023. URL: https://eprint.iacr.org/2023/122. 28

[34] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H. Shacham. Randomizable proofs and delegatable anonymous credentials. pages 108–125, 2009. doi:10.1007/978-3-642-03356-8_7. 98

[35] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046, 2018. https://eprint.iacr.org/2018/046. 118

[36] J. Blömer and J. Bobolz. Delegatable attribute-based anonymous credentials from dynamically malleable signatures. pages 221–239, 2018. doi:10.1007/978-3-319-93387-0_12. 98

[37] V. Bolgouras, A. Angelogianni, I. Politis, and C. Xenakis. Trusted and secure self-sovereign identity framework. In *Proceedings of the 17th International Conference on Availability, Reliability and Security*, ARES '22, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3538969.3544436. 29

[38] J. Bonneau, C. Herley, P. C. v. Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *2012 IEEE Symposium on Security and Privacy*, pages 553–567, 2012. doi:10.1109/SP.2012.44. 44

[39] C. Brand et al. Client to authenticator protocol (CTAP), 2019. URL: https://fidoalliance.org/specs/fido-v2.0-ps-20190130/fido-client-to-authenticator-protocol-v2.0-ps-20190130.pdf. 3, 65

[40] Brave. Enhanced privacy while browsing, 2018. URL: https://brave.com/tor-tabs-beta/. 5

[41] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. pages 315–334, 2018. doi:10.1109/SP.2018.00020. 118

[42] J. Camenisch and E. Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, page 21–30, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/586110.586114. 28

[43] C. Castelluccia et al. Betrayed by your ads! reconstructing user profiles from targeted ads. In S. Fischer-Hubner and M. Wright, editors, *Privacy Enhancing Technologies*, Lecture Notes in Computer Science. Springer, 2012. 66

[44] A. Cavoukian. Privacy by Design: The 7 Foundational Principles, Mai 2010. Revised: Oktober 2010. 143

[45] D. Cerdeira, N. Santos, P. Fonseca, and S. Pinto. Sok: Understanding the prevailing security vulnerabilities in trustzone-assisted tee systems. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1416–1432, 2020. doi:10.1109/SP40000.2020.00061. 20, 21

[46] M. Chase, S. Meiklejohn, and G. Zaverucha. Algebraic MACs and keyed-verification anonymous credentials. pages 1205–1216, 2014. doi:10.1145/2660267.2660328. 98

[47] D. Chaum. Blind signatures for untraceable payments. pages 199–203, 1982. 98

[48] D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *Advances in Cryptology*, pages 199–203, Boston, MA, 1983. Springer US. 26

[49] G. Chen, S. Chen, Y. Xiao, Y. Zhang, Z. Lin, and T. H. Lai. SgxPectre: Stealing intel secrets from SGX enclaves via speculative execution. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 142–157, Stockholm, Sweden, June 2019. IEEE. URL: https://ieeexplore.ieee.org/document/8806740/, doi:10.1109/EuroSP.2019.00020. 137

[50] S. Ciolino et al. Of two minds about two-factor: Understanding everyday FIDO u2f usability through device comparison and experience sampling. In *15th Symposium on Usable Privacy and Security (SOUPS 2019)*. USENIX Association, Aug. 2019. 87

[51] J. Colnago, L. F. Cranor, and A. Acquisti. Is there a reverse privacy paradox?

an exploratory analysis of gaps between privacy perspectives and privacy-seeking behaviors. *Proceedings on Privacy Enhancing Technologies*, 1:455–476, 2023. 145

[52] E. Commission. European digital identity architecture and reference framework – outline. Standard, European Commission, 2022. URL: https://digital-strategy.ec.europa.eu/en/library/european-digital-identity-architecture-and-reference-framework-outline. 141

[53] J. Conners, C. Devenport, S. Derbidge, N. C. Farnsworth, K. Gates, S. Lambert, C. McClain, P. Nichols, and D. Zappala. Let's authenticate: Automated certificates for user authentication. *Proceedings 2022 Network and Distributed System Security Symposium*, 2022. URL: https://api.semanticscholar.org/CorpusID:248224298. 28

[54] A. Contributors. Arkworks zkSNARK ecosystem. URL: https://arkworks.rs. 139, 164

[55] G. Couteau and M. Reichle. Non-interactive keyed-verification anonymous credentials. pages 66–96, 2019. doi:10.1007/978-3-030-17253-4_3. 98

[56] E. C. Crites and A. Lysyanskaya. Delegatable anonymous credentials from mercurial signatures. pages 535–555, 2019. doi:10.1007/978-3-030-12612-4_27. 98

[57] Ö. Dagdelen and M. Fischlin. Security analysis of the extended access control protocol for machine readable travel documents. pages 54–68, 2011. 124, 125

[58] E. Dauterman, D. Lin, H. Corrigan-Gibbs, and D. Mazières. Accountable authentication with privacy protection: The larch system for universal login. In *17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)*, pages 81–98, Boston, MA, July 2023. USENIX Association. URL: https://www.usenix.org/conference/osdi23/presentation/dauterman. 146

[59] A. Davidson, I. Goldberg, N. Sullivan, G. Tankersley, and F. Valsorda. Privacy pass: Bypassing internet challenges anonymously. 2018(3):164–180, July 2018. doi:10.1515/popets-2018-0026. 98

[60] Y. Dimova, T. Van Goethem, and W. Joosen. Everybody's looking for ssomething: A large-scale evaluation on the privacy of oauth authentication on the web. *Proceedings on Privacy Enhancing Technologies*, 4:452–467, 2023. 144

[61] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, page 265–284, Berlin, Heidelberg, 2006. Springer-Verlag. doi:10.1007/11681878_14. 24

[62] M. Dworkin. Recommendation for block cipher modes of operation: Methods for key wrapping, 2012-12-13 2012. doi:https://doi.org/10.6028/NIST.SP.800-38F. 67

[63] P. Dzurenda, R. Casanova-Marqués, L. Malina, and J. Hajny. Real-world deployment of privacy-enhancing authentication system using attribute-based credentials. In *Proceedings of the 17th International Conference on Availability, Reliability and Security*, ARES '22, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3538969.3543803. 31

[64] A. Farao, E. Veroni, C. Ntantogian, and C. Xenakis. P4g2go: A privacy-preserving scheme for roaming energy consumers of the smart grid-to-go. *Sensors*, 21(8), 2021. URL: https://www.mdpi.com/1424-8220/21/8/2686, doi:10.3390/s21082686. 29

[65] F. M. Farke, L. Lassak, J. Pinter, and M. Dürmuth. Exploring user authentication with windows hello in a small business environment. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 523–540, Boston, MA, Aug. 2022. USENIX Association. URL: https://www.usenix.org/conference/soups2022/presentation/farke. 46

[66] F. M. Farke, L. Lorenz, T. Schnitzler, P. Markert, and M. Dürmuth. "You still use the password after all" – exploring FIDO2 security keys in a small company. In *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, pages 19–35. USENIX Association, Aug. 2020. URL: https://www.usenix.org/conference/soups2020/presentation/farke. 46

[67] D. Fett, K. Yasuda, and B. Campbell. Selective Disclosure for JWTs (SD-JWT). Internet-Draft draft-ietf-oauth-selective-disclosure-jwt-05, Internet Engineering Task Force, June 2023. Work in Progress. URL: https://datatracker.ietf.org/doc/draft-ietf-oauth-selective-disclosure-jwt/05/. 27

[68] FIDO Alliance. Authenticator Level 1, 2017. URL: https://fidoalliance.org/certification/authenticator-certification-levels/authenticator-level-1/. 6, 80

[69] FIDO Alliance. Enterprise adoption best practices, 2018. URL: https://media.fidoalliance.org/wp-content/uploads/Enterprise_Adoption_Best_Practices_Lifecycle_FIDO_Alliance.pdf. 39

[70] FIDO Alliance. FIDO Security Reference, 2018. URL: https://fidoalliance.org/specs/fido-v2.0-id-20180227/fido-security-ref-v2.0-id-20180227.html. 18, 66, 68

[71] FIDO Alliance. Client to authenticator protocol (CTAP), 2021. URL: https://fidoalliance.org/specs/fido-v2.1-ps-20210615/fido-client-to-authenticator-protocol-v2.1-ps-20210615.html. 13, 39, 45

[72] FIDO Alliance. Fido authenticator lifecycle management for it administrators, 2021. URL: https://media.fidoalliance.org/wp-content/uploads/2021/04/FIDO-White-Paper-Lifecycle-Management-for-IT-Administrators.pdf. 38, 44

[73] FIDO Alliance. Consumer Habits, Trends and Adoption of Authentication Tech.

https://media.fidoalliance.org/wp-content/uploads/2022/10/Authenticate-2022-Barometer-Report.pdf, 2022. [Online; accessed 7-Sept-2023]. 11

[74] FIDO Alliance. Fido metadata statement, 2022. URL: https://fidoalliance.org/specs/mds/fido-metadata-statement-v3.0-ps-20210518.pdf. 39

[75] FIDO Alliance. Fido security reference, 2022. URL: https://fidoalliance.org/specs/common-specs/fido-security-ref-v2.1-ps-20220523.html. 43

[76] FIDO Alliance. Multi-device fido credentials, 2022. URL: https://media.fidoalliance.org/wp-content/uploads/2022/03/How-FIDO-Addresses-a-Full-Range-of-Use-Cases-March24.pdf. 43, 45

[77] M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In C. Cachin and J. L. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, pages 1–19, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. 24

[78] N. Frymann, D. Gardham, and M. Manulis. Unlinkable delegation of webauthn credentials. *IACR Cryptol. ePrint Arch.*, page 303, 2022. URL: https://eprint.iacr.org/2022/303. 42

[79] J. Garbis and J. W. Chapman. *Privileged Access Management*, pages 155–161. Apress, Berkeley, CA, 2021. doi:10.1007/978-1-4842-6702-8_12. 42

[80] C. Garman, M. Green, and I. Miers. Decentralized anonymous credentials. 2014. 98

[81] G. M. Garrido, X. Liu, F. Matthes, and D. Song. Lessons learned: Surveying the practicality of differential privacy in the industry. *arXiv preprint arXiv:2211.03898*, 2022. 144

[82] D. Genkin et al. Synesthesia: Detecting screen content via remote acoustic side channels. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 853–869, 2019. doi:10.1109/SP.2019.00074. 85

[83] S. Ghorbani Lyastani, M. Schilling, M. Neumayr, M. Backes, and S. Bugiel. Is fido2 the kingslayer of user authentication? a comparative usability study of fido2 passwordless authentication. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 268–285, 2020. doi:10.1109/SP40000.2020.00047. 35, 36, 87

[84] O. Goga et al. Exploiting innocuous activity for correlating users across sites. In *22nd International World Wide Web Conference, WWW '13*, pages 447–458, 2013. 66

[85] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 218–229, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/28395.28420. 25

[86] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, page 291–304, New York, NY, USA, 1985. Association for Computing Machinery. doi:10.1145/22145.22178. 26

[87] Google. Chromium: implementation of fido2 client. URL: https://chromium.googlesource.com/chromium/src/+/refs/heads/main/device/fido/get_assertion_task.cc. 71

[88] Google. Opensk: open-source implementation for fido u2f and fido 2 security keys. URL: https://github.com/google/OpenSK/blob/5e682d9e176e936c22fcb963a708ffb0b47a33e6/src/ctap/mod.rs. 73, 158

[89] Google. Play Integrity API. URL: https://developer.android.com/google/play/integrity/overview. 164

[90] I. Gordin, A. Graur, S. Vlad, and C. I. Adomniței. Moving forward passwordless authentication: challenges and implementations for the private cloud. In *2021 20th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, pages 1–5, 2021. doi:10.1109/RoEduNet54112.2021.9638271. 46

[91] E. Grosse and M. Upadhyay. Authentication at scale. *IEEE Security and Privacy*, 11(1):15–22, 2013. doi:10.1109/MSP.2012.162. 41

[92] J. Groth. On the size of pairing-based non-interactive arguments. pages 305–326, 2016. doi:10.1007/978-3-662-49896-5_11. 118, 139, 164

[93] C. Guo, Q. Cai, Q. Wang, and J. Lin. Extending registration and authentication processes of fido2 external authenticator with qr codes. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 518–529, 2020. doi:10.1109/TrustCom50675.2020.00076. 119, 168

[94] S. Gupta, A. Singhal, and A. Kapoor. A literature survey on social engineering attacks: Phishing attack. In *2016 International Conference on Computing, Communication and Automation (ICCCA)*, pages 537–540, 2016. doi:10.1109/CCAA.2016.7813778. 12

[95] M. J. Haber. *Privileged Access Management*, pages 151–171. Apress, Berkeley, CA, 2020. doi:10.1007/978-1-4842-5914-6_11. 42

[96] M. J. Haber and B. Hibbert. *Shared User Credentials*, pages 25–38. Apress, Berkeley, CA, 2018. doi:10.1007/978-1-4842-3048-0_2. 42

[97] T. Hackenjos, B. Wagner, J. Herr, J. Rill, M. Wehmer, N. Goerke, and I. Baumgart. Fido2 with two displays-or how to protect security-critical web transactions against malware attacks, 2022. arXiv:2206.13358. 29

[98] T. Hanrahan. Analysis of windows cardspace identity management system. 2011. URL: https://api.semanticscholar.org/CorpusID:62928380. 16

[99] L. Hanzlik, J. Loss, and B. Wagner. Token meets wallet: Formalizing privacy and revocation for FIDO2. In *2023 IEEE Symposium on Security and Privacy (IEEE SP)*, pages 978–995, Los Alamitos, CA, USA, May 2023. IEEE Computer Society. URL: https://doi.ieeecomputersociety.org/10.1109/SP46215.20

23.00056, doi:10.1109/SP46215.2023.00056. 18, 35, 37, 101, 102, 105, 106, 107, 115, 116, 125, 127

[100] D. Hardt. The OAuth 2.0 Authorization Framework. RFC 6749, Oct. 2012. URL: https://www.rfc-editor.org/info/rfc6749, doi:10.17487/RFC6749. 27

[101] haveibeenpwned.com. "pwned passwords" dataset. URL: https://haveibeenpwned.com/Passwords. 11

[102] A. Heinrich, M. Hollick, T. Schneider, M. Stute, and C. Weinert. PrivateDrop: Practical Privacy-Preserving authentication for apple AirDrop. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3577–3594. USENIX Association, Aug. 2021. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/heinrich. 30

[103] S. Hendrickson, J. Iyengar, T. Pauly, S. Valdez, and C. A. Wood. Rate-limited token issuance protocol. Internet-Draft draft-privacypass-rate-limit-tokens-03, IETF Secretariat, July 2022. 98

[104] Internation Civil Aviation Organization. Doc 9303 Machine readable travel documents - Part 10: Logical data structure (LDS) for storage of biometrics and other data in the contactless integrated circuit (IC). Technical report, ICAO, 2021. URL: https://www.icao.int/publications/documents/9303_p10_cons_en.pdf. 122, 123

[105] Internet Engineering Task Force (IETF). JSON Web Token (JWT). https://datatracker.ietf.org/doc/html/rfc7519. [Online; accessed 7-Sept-2023]. 27

[106] ISO Central Secretary. Information technology — security techniques — entity authentication assurance framework. Standard ISO/IEC 29115, International Organization for Standardization, Geneva, CH, 2013. URL: https://www.iso.org/standard/45138.html. 11

[107] ISO Central Secretary. Personal identification — ISO-compliant driving licence — Part 5: Mobile driving licence (mDL) application. Standard ISO/IEC 18013-5:2021, International Organization for Standardization, Geneva, CH, 2021. URL: https://www.iso.org/standard/69084.html. 141

[108] ISO Central Secretary. Information security, cybersecurity and privacy protection — biometric information protection. Standard ISO/IEC 24745:2022, International Organization for Standardization, Geneva, CH, 2022. URL: https://www.iso.org/standard/75302.html. 27

[109] P. Jagwani and S. Kaushik. Defending location privacy using zero knowledge proof concept in location based services. In *2012 IEEE 13th International Conference on Mobile Data Management*, pages 368–371, 2012. doi:10.1109/MDM.2012.23. 26

[110] S. P. Jim Banoczi, Harry Perper. NIST privileged account management: Securing privileged accounts for the financial services sector. Technical report, National Institute of Standards and Technology, 2017. 42

[111] D. Johnson et al. The elliptic curve digital signature algorithm (ecdsa). *Int. J. Inf. Sec.*, 1(1), 2001. 14

[112] R. P. Jover. Security analysis of sms as a second factor of authentication. *Commun. ACM*, 63(12):46–52, nov 2020. doi:10.1145/3424260. 12

[113] S. Karunakaran et al. Data breaches: User comprehension, expectations, and concerns with handling exposed data. In *Fourteenth Symposium on Usable Privacy and Security, SOUPS 2018, Baltimore, MD, USA, August 12-14, 2018*, pages 217–234, 2018. 65

[114] E. Klieme, J. Wilke, N. van Dornick, and C. Meinel. Fidonuous: A fido2/webauthn extension to support continuous web authentication. In *2020*

IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pages 1857–1867, 2020. doi:10.1109/TrustCom50675.2020.00254. 29, 43

[115] E. Kovacs. Cisco hacked by ransomware gang, data stolen, August 2022. URL: https://www.securityweek.com/cybercriminals-breached-cisco-systems-and-stole-data. 3, 12

[116] E. Kovacs. Hackers stole source code, personal data from dropbox following phishing attack, November 2022. URL: https://www.securityweek.com/hackers-stole-source-code-personal-data-dropbox-following-phishing-attack. 3, 12

[117] E. Kovacs. High-profile hacks show effectiveness of mfa fatigue attacks, September 2022. URL: https://www.securityweek.com/high-profile-hacks-show-effectiveness-mfa-fatigue-attacks. 3, 12

[118] S. Krenn, T. Lorünser, A. Salzer, and C. Striecks. Towards attribute-based credentials in the cloud. pages 179–202, 2017. doi:10.1007/978-3-030-02641-7_9. 98

[119] K. Krombholz, W. Mayer, M. Schmiedecker, and E. Weippl. "i have no idea what i'm doing" - on the usability of deploying HTTPS. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1339–1356, Vancouver, BC, Aug. 2017. USENIX Association. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/krombholz. 46

[120] K. Krombholz, W. Mayer, M. Schmiedecker, and E. R. Weippl. "I have no idea what I'm doing" - on the usability of deploying HTTPS. pages 1339–1356, 2017. 34, 165

[121] A. Kumar et al. Web authentication: An API for accessing public key credentials - level 2. W3C recommendation, W3C, Apr. 2021. URL: https://www.w3.org/TR/2021/REC-webauthn-2-20210408/. 65

[122] M. Kumar, S. Chand, and C. P. Katti. A secure end-to-end verifiable internet-voting system using identity-based blind signature. *IEEE Systems Journal*, 14(2):2032–2041, 2020. doi:10.1109/JSYST.2019.2940474. 26

[123] J. Kunke, S. Wiefling, M. Ullmann, and L. L. Iacono. Evaluation of account recovery strategies with fido2-based passwordless authentication. 2021. URL: https://arxiv.org/abs/2105.12477, doi:10.48550/ARXIV.2105.12477. 44

[124] S. Kurowski, R. Litwing, and G. Lückemeyer. A view on iso/iec 27001 compliant identity lifecycles for it service providers. In *2015 World Congress on Internet Security (WorldCIS)*, pages 85–90, 2015. doi:10.1109/WorldCIS.2015.7359420. 38

[125] S. Lab. C++ library for zkSNARKs. URL: https://github.com/scipr-lab/libsnark. 139, 164

[126] J. Lazar, J. H. Feng, and H. Hochheiser. Chapter 5 - surveys. In J. Lazar, J. H. Feng, and H. Hochheiser, editors, *Research Methods in Human Computer Interaction (Second Edition)*, pages 105–133. Morgan Kaufmann, Boston, second edition edition, 2017. URL: https://www.sciencedirect.com/science/article/pii/B9780128053904000054, doi:https://doi.org/10.1016/B978-0-12-805390-4.00005-4. 47

[127] R. Lindemann and E. Tiffany. FIDO UAF protocol specification, 2020. URL: https://fidoalliance.org/specs/fido-uaf-v1.2-ps-20201020/fido-uaf-protocol-v1.2-ps-20201020.pdf. 65

[128] B. Liu, X. Zhang, R. Shi, M. Zhang, and G. Zhang. Sepsi: A secure and efficient privacy-preserving set intersection with identity authentication in iot. *Mathematics*, 10(12), 2022. URL: https://www.mdpi.com/2227-7390/10/12/2120, doi:10.3390/math10122120. 24

[129] T. Looker, V. Kalos, A. Whitehead, and M. Lodder. OpenID Connect Core 1.0.

Technical report, OpenID Foundation. URL: https://openid.net/specs/openid-connect-core-1_0.html. 27

[130] T. Looker, V. Kalos, A. Whitehead, and M. Lodder. The BBS Signature Scheme. Internet-Draft draft-irtf-cfrg-bbs-signatures-03, Internet Engineering Task Force, July 2023. Work in Progress. URL: https://datatracker.ietf.org/doc/draft-irtf-cfrg-bbs-signatures/03/. 28

[131] D. Maram, H. Malvai, F. Zhang, N. Jean-Louis, A. Frolov, T. Kell, T. Lobban, C. Moy, A. Juels, and A. Miller. Candid: Can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1348–1366, 2021. doi:10.1109/SP40001.2021.00038. 146

[132] Microsoft. Understanding the Remote Desktop Protocol (RDP). https://learn.microsoft.com/en-us/troubleshoot/windows-server/remote/understanding-remote-desktop-protocol. [Online; accessed 7-Sept-2023]. 41

[133] Microsoft. Microsoft digital defense report 2022, 2022. URL: https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE5bUvv. 3, 34

[134] G. Moore. Crossing the chasm: Marketing and selling disruptive products to mainstream customers., 2002. 33

[135] K. S. Murugiah Souppaya. NIST special publication 800-46r2 guide to enterprise telework, remote access, and bring your own device (byod) security. Technical Report NIST Special Publication (SP) 800-46r2, National Institute of Standards and Technology, 2016. doi:10.6028/NIST.SP.800-46r2. 38, 40

[136] A. Narayanan et al. On the feasibility of internet-scale author identification. In *IEEE Symposium on Security and Privacy, SP 2012, 21-23 May 2012, San Francisco, California, USA*, pages 300–314. IEEE Computer Society, 2012. doi:10.1109/SP.2012.46. 66

[137] Neal Mueller. Credential stuffing. https://owasp.org/www-community/attacks/Credential_stuffing. [Online; accessed 7-Sept-2023]. 12

[138] Network Working Group. The Secure Shell (SSH) Protocol Architecture. https://datatracker.ietf.org/doc/html/rfc4251. [Online; accessed 7-Sept-2023]. 41

[139] NIST. Digital Identity Guidelines Federation and Assertions. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63C-4.ipd.pdf. [Online; accessed 7-Sept-2023]. 42

[140] NIST. Glossary: least privilege. https://csrc.nist.gov/glossary/term/least_privilege. [Online; accessed 7-Sept-2023]. 42

[141] NIST. Glossary: pharming. https://csrc.nist.gov/glossary/term/pharming. [Online; accessed 7-Sept-2023]. 12

[142] NIST. Glossary: phishing. https://csrc.nist.gov/glossary/term/phishing. [Online; accessed 7-Sept-2023]. 12

[143] NIST. Glossary: replay. https://csrc.nist.gov/glossary/term/replay_attack. [Online; accessed 7-Sept-2023]. 12

[144] NIST. Security Requirements for Cryptographic Modules . https://csrc.nist.gov/pubs/fips/140-2/upd2/final. [Online; accessed 7-Sept-2023]. 37

[145] OECD. Emerging privacy-enhancing technologies. (351), 2023. URL: https://www.oecd-ilibrary.org/content/paper/bf121be4-en, doi:https://doi.org/https://doi.org/10.1787/bf121be4-en. 27

[146] Y. Okawa, S. Yamaguchi, H. Gomi, and T. Uehara. Implementation of an extended fido2 authenticator using attribute-based signatures. In *2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pages 825–832, 2021. doi:10.1109/QRS-C55045.2021.00125. 30, 119, 168

[147] OpenID Foundation. Government-issued digital credentials and the privacy landscape, 2023. URL: https://openid.net/wordpress-content/uploads/2023/04/DRAFT-Government-issued-Digital-Credentials-and-the-Privacy-Landscape-publiccomment.pdf. 27

[148] OWASP. Brute Force Attack. https://owasp.org/www-community/attacks/Brute_force_attack. [Online; accessed 7-Sept-2023]. 11

[149] K. Owens, O. Anise, A. Krauss, and B. Ur. User perceptions of the usability and security of smartphones as FIDO2 roaming authenticators. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*, pages 57–76. USENIX Association, Aug. 2021. URL: https://www.usenix.org/conference/soups2021/presentation/owens. 35, 36

[150] K. Papadamou, M. Charalambous, P. Papagiannis, M. Sirivianos, C. Xenakis, V. Bolgouras, A. Tsiota, D. Savva, E. Kotsifakos, G. Gugulea, S. C. Teican, P. Scurtu, I. Stroinea, K. Samari, and C. Segura. Identity verification with privacy-preserving credentials for anonymous access to online services. 2020. 29

[151] C. Paquin and G. Zaverucha. U-prove technology overview. Technical report, Microsoft. URL: https://github.com/microsoft/uprove-node-reference/blob/main/doc/U-ProveCryptographicSpecificationV1.1Revision5.pdf. 28

[152] E. Parliament. Regulation (EU) 2019/1157 of the European Parliament and of the Council of 20 june 2019 on strengthening the security of identity cards of union citizens and of residence documents issued to union citizens and their family members exercising their right of free movement (text with EEA relevance.), Jul 2019. 97

[153] J. F. Paul Grassi, Michael Garcia. NIST special publication 800-63b digital identity guidelines authentication and lifecycle management. Technical Report NIST Special Publication (SP) 800-63B, National Institute of Standards and Technology, 2017. doi:10.6028/NIST.SP.800-63-3. 3, 10, 37, 39, 40

[154] K. Pfeffer, A. Mai, A. Dabrowski, M. Gusenbauer, P. Schindler, E. Weippl, M. Franz, and K. Krombholz. On the usability of authenticity checks for hardware security tokens. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 37–54. USENIX Association, Aug. 2021. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/pfeffer. 38

[155] A. Pfitzmann and M. Hansen. Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management–a consolidated proposal for terminology. *Version v0*, 31, 01 2007. 146

[156] A. Pfitzmann and M. Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. *URL: http://dud. inf. tu-dresden. de/literatur/Anon*$_T$*erminology*$_v$0, 34, 012010. 22

[157] E. S. Philipp Markert, Maximilian Golla and M. Durmuth. A comparative long-term study of fallback authentication. 2019. URL: https://www.ndss-symposium.org/wp-content/uploads/2019/02/usec2019_04-4_Markert_paper.pdf, doi:https://dx.doi.org/10.14722/usec.2019.23030. 44

[158] F. Putz, S. Schön, and M. Hollick. Future-proof web authentication: Bring your own fido2 extensions. In A. Saracino and P. Mori, editors, *Emerging Technologies for Authorization and Authentication*, pages 17–32, Cham, 2021. Springer International Publishing. 119, 168

[159] H. Ragab, A. Milburn, K. Razavi, H. Bos, and C. Giuffrida. CrossTalk: Speculative data leaks across cores are real. pages 1852–1867, 2021. doi:10.1109/SP40001.2021.00020. 137

[160] R. Ranjan. Password spraying attack, 2022. URL: https://owasp.org/www-community/attacks/Password_Spraying_Attack. 3

[161] J. Reynolds, N. Samarin, J. Barnes, T. Judd, J. Mason, M. Bailey, and S. Egelman. Empirical measurement of systemic 2FA usability. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 127–143. USENIX Association, Aug. 2020. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/reynolds. 44

[162] C. J. Riederer et al. Linking users across domains with location data: Theory and validation. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016*, pages 707–719, 2016. 66

[163] Rishu Ranjan. Password Spraying Attack. https://owasp.org/www-community/attacks/Password_Spraying_Attack. [Online; accessed 7-Sept-2023]. 11

[164] R. L. Rivest, L. Adleman, and M. L. Dertouzos. On data banks and privacy homomorphisms. In R. A. DeMillo, D. P. Dobkin, A. K. Jones, and R. J. Lipton, editors, *Foundations of Secure Computation*, pages 165–179. Academic Press. 25

[165] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, feb 1978. doi:10.1145/359340.359342. 14

[166] S. Rose, O. Borchert, S. Mitchell, and S. Connelly. NIST zero trust architecture. Technical report, National Institute of Standards and Technology, 2020. 34

[167] M. Rosenberg, J. White, C. Garman, and I. Miers. `zk-creds`: Flexible anonymous credentials from zkSNARKs and existing identity infrastructure. Cryptology ePrint Archive, Report 2022/878, 2022. https://eprint.iacr.org/2022/878. 99

[168] L. Roy, S. Lyakhov, Y. Jang, and M. Rosulek. Practical Privacy-Preserving authentication for SSH. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 3345–3362, Boston, MA, Aug. 2022. USENIX Association. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/roy. 30

[169] D. Saxe. Account recovery. *IDPro*, 2021. URL: https://www.sciencedirect.com/science/article/pii/S0167404821003114. 44

[170] F. Schwarz, K. Do, G. Heide, L. Hanzlik, and C. Rossow. Feido: Recoverable FIDO2 tokens using electronic ids. In *29th ACM Conference on Computer and Communications Security (CCS)*. Association for Computing Machinery, November 2022. 30, 99

[171] A. Sharif, M. Ranzi, R. Carbone, G. Sciarretta, F. A. Marino, and S. Ranise. The eidas regulation: A survey of technological trends for european electronic identity schemes. *Applied Sciences*, 12(24), 2022. URL: https://www.mdpi.com/2076-3417/12/24/12679, doi:10.3390/app122412679. 11

[172] SoloKeys. Solokeys: open-source firmware implementation. URL: https://github.com/solokeys/solo/blob/master/fido2/crypto.c. 90, 158

[173] A. Sonnino, M. Al-Bassam, S. Bano, S. Meiklejohn, and G. Danezis. Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers. 2019. 98

[174] S. Srinivas, D. Balfanz, E. Tiffany, and A. Czeskis. Universal 2nd factor (U2F) overview, 2017. URL: https://fidoalliance.org/specs/fido-u2f-v1.2-ps-20170411/fido-u2f-overview-v1.2-ps-20170411.pdf. 65

[175] G. N. Stock and M. V. Tatikonda. External technology integration in product and process development. *International Journal of Operations & Production Management*, 24(7):642–665, Jan 2004. doi:10.1108/01443570410541975. 35

[176] V. Sucasas, A. Aly, G. Mantas, J. Rodriguez, and N. Aaraj. Secure multi-party computation-based privacy-preserving authentication for smart cities. *IEEE Transactions on Cloud Computing*, pages 1–18, 2023. doi:10.1109/TCC.2023.3294621. 25

[177] M. Tahaei, T. Li, and K. Vaniea. Understanding privacy-related advice on stack overflow. *Proc. Priv. Enhancing Technol.*, 2022(2):114–131, 2022. 144

[178] K.-L. Tan, C.-H. Chi, and K.-Y. Lam. Secure and privacy-preserving sharing of personal health records with multi-party pre-authorization verification. *Wireless Networks*, Sep 2022. doi:10.1007/s11276-022-03114-6. 25

[179] D. Temoshok, J. Fenton, Y.-Y. Choong, N. Lefkovitz, A. Regenscheid, and J. Richer. NIST digital identity guidelines: Authentication and lifecycle management. Technical report, National Institute of Standards and Technology, 2022. 38

[180] Thales. The electronic passport in 2021 and beyond, 2021. URL: https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/passport/electronic-passport-trends. 97

[181] K. Thomas et al. Data breaches, phishing, or malware?: Understanding the risks of stolen credentials. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017*, pages 1421–1434, 2017. 65

[182] K. Thomas, F. Li, A. Zand, J. Barrett, J. Ranieri, L. Invernizzi, Y. Markov, O. Comanescu, V. Eranti, A. Moscicki, D. Margolis, V. Paxson, and E. Bursztein, editors. *Data breaches, phishing, or malware? Understanding the risks of stolen credentials*, 2017. 11

[183] F. Tramèr, D. Boneh, and K. Paterson. Remote Side-Channel attacks on anonymous transactions. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 2739–2756. USENIX Association, Aug. 2020. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/tramer. 147

[184] E. Ulqinaku, H. Assal, A. Abdou, S. Chiasson, and S. Capkun. Is real-time phishing eliminated with FIDO? social engineering downgrade attacks against FIDO protocols. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3811–3828. USENIX Association, Aug. 2021. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/ulqinaku. 12, 45

[185] C. Utz, S. Amft, M. Degeling, T. Holz, S. Fahl, and F. Schaub. Privacy rarely considered: Exploring considerations in the adoption of third-party services by websites, 2022. arXiv:2203.11387. 144

[186] D. K. Vallabhadas, M. Sandhya, S. Sarkar, and Y. R. Chandra. Multimodal biometric authentication using fully homomorphic encryption. In *2023 2nd International Conference on Paradigm Shifts in Communications Embedded Systems, Machine Learning and Signal Processing (PCEMS)*, pages 1–6, 2023. doi:10.1109/PCEMS58491.2023.10136104. 25

[187] A. Vasileios Grammatopoulos, I. Politis, and C. Xenakis. A web tool for analyzing fido2/webauthn requests and responses. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, ARES 21, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3465481.3469209. 46

[188] Verizon. Insider threat report, 2019. URL: https://www.verizon.com/business/resources/reports/insider-threat-report.pdf. 2

[189] W3C. Web authentication: An API for accessing public key credentials level 2, 2021. URL: ttps://www.w3.org/TR/webauthn-2/. 13, 17

[190] P. Wagner, K. Heid, and J. Heider. Remote webauthn: Fido2 authentication for less accessible devices, 2021. URL: https://publica.fraunhofer.de/handle/publica/410137, doi:10.5220/0010192703680375. 29

[191] D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang. Targeted online password guessing: An underestimated threat. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 1242–1254, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2976749.2978339. 11

[192] Y. Wang, Q. Huang, H. Li, M. Xiao, S. Ma, and W. Susilo. Private set intersection with authorization over outsourced encrypted datasets. *IEEE Transactions on*

Information Forensics and Security, 16:4050–4062, 2021. doi:10.1109/TIFS.2021.3101059. 24

[193] J. Weidman and J. Grossklags. I like it, but i hate it: Employee perceptions towards an institutional transition to byod second-factor authentication. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, ACSAC '17, page 212–224, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3134600.3134629. 46

[194] T. Whalen, T. Meunier, M. Kodali, A. Davidson, M. Fayed, A. Faz-Hernández, W. Ladd, D. Maram, N. Sullivan, B. C. Wolters, M. Guerreiro, and A. Galloni. Let the right one in: Attestation as a usable CAPTCHA alternative. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 599–612, Boston, MA, Aug. 2022. USENIX Association. URL: https://www.usenix.org/conference/soups2022/presentation/whalen. 30

[195] D. Winder. Uber hack update: Was sensitive user data stolen & did 2fa open door to hacker?, September 2022. URL: https://www.forbes.com/sites/daveywinder/2022/09/18/has-uber-been-hacked-company-investigates-cybersecurity-incident-as-law-enforcement-alerted/?sh=7463a17d6056. 3, 12

[196] X. Yao, R. Zhang, and Y. Zhang. Differential privacy-preserving user linkage across online social networks. In *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, pages 1–10, 2021. doi:10.1109/IWQOS52092.2021.9521333. 24

[197] Yubico. Enterprise Attestation. https://developers.yubico.com/WebAuthn/Concepts/Enterprise_Attestation/. [Online; accessed 7-Sept-2023]. 39

[198] ZKProof. ZKProof Community Reference. https://docs.zkproof.org/reference.pdf, 2022. [Online; accessed 7-Sept-2023]. 7