A-WAYF: Automated Where Are You From in Multilateral Federations

Erwin Kupris[1,*,†], Tobias Hilbig[1,†], David Pierre Sugar[1,†] and Thomas Schreck[1,†]

[1]*Munich University of Applied Sciences HM, Munich, Germany*

Abstract

In recent years, web authentication has seen significant simplification for users with the widespread adoption of social logins like "Login with Google". The technology behind this trend is Federated Identity Management (FIM), which allows users to access various online services using credentials issued by another trusted organization. Instead of relying on a single identity provider, services can often be accessed through multiple trusted ones, especially in the research and education sector. However, the process of identifying the appropriate identity provider, i.e., the user's home organization, has been a long-standing challenge within federated environments. With third-party cookies soon to be deprecated, existing solutions for automating this process will lose their technical foundation. Moreover, these solutions require manual selection by users during their initial visit to a website. In this paper, we propose A-WAYF, a fully automated, technology-independent solution to this problem. Instead of using cookies, we store information about the user's identity provider alongside their credential. In addition, the browser mediates between all parties to ensure user privacy and security. We demonstrate our solution's feasibility through a proof of concept based on passkeys and OpenID Federation. Finally, we propose extensions for all protocols used in the process. Our results show that only minimal changes to existing protocols are required to automate this process. Building upon emerging technologies, we pave the way to a vastly improved user experience when interacting with federated services.

Keywords

Federation, FIM, IdP Discovery, OpenID Federation, SAML 2.0, FIDO2, CTAP2, WebAuthn

1. Introduction

Logging into websites has recently become increasingly simplified. Social logins such as "Login with Google" are prevalent across many online services. This authentication option benefits all involved parties: Users no longer need to create new accounts for each website, while websites can delegate the complex tasks of user identification and authentication to an Identity Provider (IdP). This process is called Single Sign-On (SSO) and can be seen as a special case of Federated Identity Management (FIM), in which only a single IdP exists. Long before social logins became ubiquitous, FIM was used in the Research and Education (R&E) sector to build complex, multilateral federations. These federations enable seamless collaboration between

TDI 2024: 2nd International Workshop on Trends in Digital Identity, April 9, 2024, Rome, Italy

*Corresponding author.

†These authors contributed equally.

erwin.kupris@hm.edu (E. Kupris); tobias.hilbig@hm.edu (T. Hilbig); sugar@hm.edu (D. P. Sugar); thomas.schreck@hm.edu (T. Schreck)

0000-0002-2799-5197 (E. Kupris); 0000-0002-2904-4758 (T. Hilbig); 0009-0007-0056-602X (D. P. Sugar); 0000-0002-8960-6986 (T. Schreck)

institutions by allowing users to access resources of other member institutions using their home credentials. Additionally, the global R&E inter-federation eduGAIN connects national federations, enabling even broader collaboration possibilities. Within such an architecture, a single Service Provider (SP) might be accessible through credentials from numerous IdPs [1].

A crucial part of FIM is the Discovery Service (DS), also called Where Are You From (WAYF), which allows users to find and select an organization to be used for authentication. For simple scenarios with a small number of compatible organizations, such as social logins, buttons specific to the organization can be used. Users simply click the button representing the respective organization, and their browser is redirected to the appropriate IdP. However, this approach does not scale for large federations that support numerous IdPs. Other DS solutions often require manual intervention by the user, e.g., selecting their organization from a list of supported ones or entering their email address. Therefore, in this work, we investigate how the WAYF process in multilateral, federated environments can be automated while preserving user privacy. To realize this kind of automation even during the user's initial visit to an SP, we propose to store affiliation information alongside the user's credential.

This work offers three contributions: We present "A-WAYF", an automated WAYF solution that is both user-friendly and privacy preserving, a Proof of Concept (PoC) implementation of A-WAYF using passkeys and OpenID Federation, and ideas for extensions to adapt existing protocols to support the functionality required by A-WAYF.

The remainder of this paper is structured as follows: In Section 2, background information about federations, the WAYF process, and FIDO2 is provided. Section 3 presents research related to our work. We present our solution in Section 4 and our PoC in Section 5. The proposed extensions, security and privacy considerations, and avenues for future work are discussed in Section 6. The paper concludes with Section 7.

2. Background

We briefly discuss the protocols used for communication in federated environments and provide information on existing DS solutions. In addition, we present the FIDO2 framework because our PoC implementation relies on passkeys.

2.1. Federation Protocols

Secure communication between members of a federation is enabled by specific protocols. The Security Assertion Markup Language V2.0 (SAML 2.0) [2] is a well-established and widely adopted protocol for exchanging authentication and authorization data, allowing users to access resources across federated domains. OpenID Connect (OIDC) [3] builds upon OAuth 2.0 [4] to provide a user-friendly authentication layer, whereas OAuth focuses on authorization by enabling secure access to resources without exposing user credentials. While they are employed in SSO and social login implementations, OIDC and OAuth are not compatible with complex federations. However, the OpenID Federation draft [5] extends OIDC and OAuth to support multilateral federation scenarios through a hierarchical architecture. It defines mechanisms for discovering federation entities and dynamically resolving metadata, allowing IdPs and SPs to establish trust and share information across federated domains.

2.2. Discovery Services

The first step in a federated authentication procedure is usually the selection of the user's organization from a list of supported ones. In R&E federations, this step is often realized through a DS. At a high level, the DS is responsible for collecting metadata from the supported IdPs, listing the respective organizations in a user dialog, collecting the user's selection, and redirecting the user's browser to the appropriate endpoint of the selected IdP. The DS can either be embedded into the SP or run by a trusted external entity. The latter option requires an additional browser redirect to the external DS. Various DSs are deployed in R&E federations today, for example, the SeamlessAccess service [6]. Many of these DS solutions rely on manual interaction by the user. For instance, users may be required to select their home organization from a list, which can be extensive, or they might be prompted to enter their organization's name. Moreover, the user's selection can be stored in their browser, either as cookies or local storage entries. This would allow the DS to recommend the previously chosen IdP when the user returns to the same SP or visits another SP linked to the same DS. Another approach, the WebFinger protocol [7], infers information such as the IdP responsible for a domain or user, but still requires manual input. Some services rely on whitelisting the IP ranges of R&E institutions for authentication. However, this method is infeasible for scenarios in which users access services from various locations or through VPNs.

2.3. FIDO2

The FIDO2 framework combines the W3C Web Authentication (WebAuthn) standard [8] and the FIDO Alliance's Client To Authenticator Protocol 2 (CTAP2) [9]. WebAuthn manages communication between the client, usually a browser, and the Relying Party (RP), usually a web server. CTAP2 facilitates interaction between the client and the authenticator, which can be a roaming security key, a device's platform authenticator, or a combination of both like a smartphone-based authenticator. FIDO2 employs asymmetric cryptography and a challenge-response procedure for user authentication. When using FIDO2 for a service, a unique key-pair is created for an account and bound to the RP's domain. The private key usually resides on the authenticator, whereas the public key is stored by the RP. During authentication, the RP sends a challenge to the authenticator. This challenge is signed with the corresponding private key. The RP verifies the signature using the stored public key. FIDO2 presents a framework for phishing-resistant authentication on the web and is currently gaining support from device and browser vendors [10]. As a result, websites can now offer a secure, user-friendly alternative to passwords that mitigates most of their shortcomings.

3. Related Work

The challenge of IdP discovery has persisted for nearly two decades [11]. An approach to automate the WAYF process was presented by Kataoka et al. in 2009 [12], which extends a centralized, external DS. Their solution relies on TLS client authentication to match the organization name in client certificates with corresponding values in DS metadata. However, this method poses security risks, potentially leaking user affiliation and entire client certificates

to malicious DS instances. In contrast, A-WAYF dynamically resolves trust relationships between SPs and IdPs, eliminating this vulnerability. Additionally, A-WAYF provides a more flexible and user-friendly solution while offering compatibility with various authentication protocols and platforms, ensuring broader accessibility and ease of implementation.

To the best of our knowledge, no further academic publications exist in which the IdP discovery process is fully automated. Therefore, we present several established as well as proposed solutions and show how they differ from ours.

The SAML 2.0 protocol includes an IdP Discovery Profile that enables an SP to automatically infer the user's IdP or offer relevant suggestions [11]. In a domain common to the federated entities, the set of IdPs previously visited by the user is stored in a cookie. This list of potential IdPs is presented to the user for selection before redirecting the browser to the chosen IdP. However, the scalability and flexibility of this approach are limited because of the extensive static configuration required for the common domain. This profile is complemented by the SAML 2.0 IdP Discovery Service Protocol and Profile [13] proposing an alternative solution through a centralized DS. The DS can incorporate features like an IdP persistence service, i.e., storing the chosen IdP in the browser, and a metadata query service for receiving up-to-date SAML 2.0 metadata. Both SAML 2.0 profiles can be integrated into SPs. SeamlessAccess [6], the state-of-the-art DS in the R&E sector, enhances user experience (UX) but still requires manual organization selection in the absence of a browser state. Furthermore, its automation approach relies on soon-to-be deprecated third-party cookies, posing a risk to its technical foundation [14]. A-WAYF resolves these limitations by not relying on cookies and dynamically determining the user's affiliation. This results in an automated WAYF process that even works the first time a user visits the SP.

The Credential Management API [15], in Working Draft status at W3C, offers browsers a standardized method for creating and accessing credentials. Five types of credentials are specified: "OTPCredential" and "PasswordCredential" are not relevant for this work. "PublicKeyCredential" corresponds to WebAuthn credentials as described in Section 2. The "FederatedCredential" type differs from other types by also storing the IdP and the federation protocol with the credential. It can be used by services to programmatically store and retrieve credentials issued by federated IdPs within the browser's credential storage, which is the main difference to our solution. Browser support for this credential type is limited, and to the best of our knowledge, there are no prominent implementations of this API on the Internet.

The last credential type is called "IdentityCredential". It is specified in the Federated Credential Management API (FedCM) [16] and was proposed in response to the imminent deprecation of third-party cookies, which poses a threat to the functionality of social logins. By transitioning the information about the user's logged-in status from a third-party cookie to a dynamic process mediated by the browser, FedCM circumvents these restrictions. It also provides a user-friendly interface for selecting an account when signing in to a service. While our solution aligns well with most goals, privacy concerns, and UX considerations addressed by FedCM, it is important to note a distinction. Our focus lies on multilateral federations, whereas FedCM primarily addresses bilateral federations and enhances the social login process. It assumes that the SP offers a limited number of supported IdPs, allowing the browser to query each IdP repeatedly. While discussions about aligning FedCM's functionality with the use-cases of R&E federations are ongoing, their proposals [17, 18] do not include ways to automate the WAYF process.

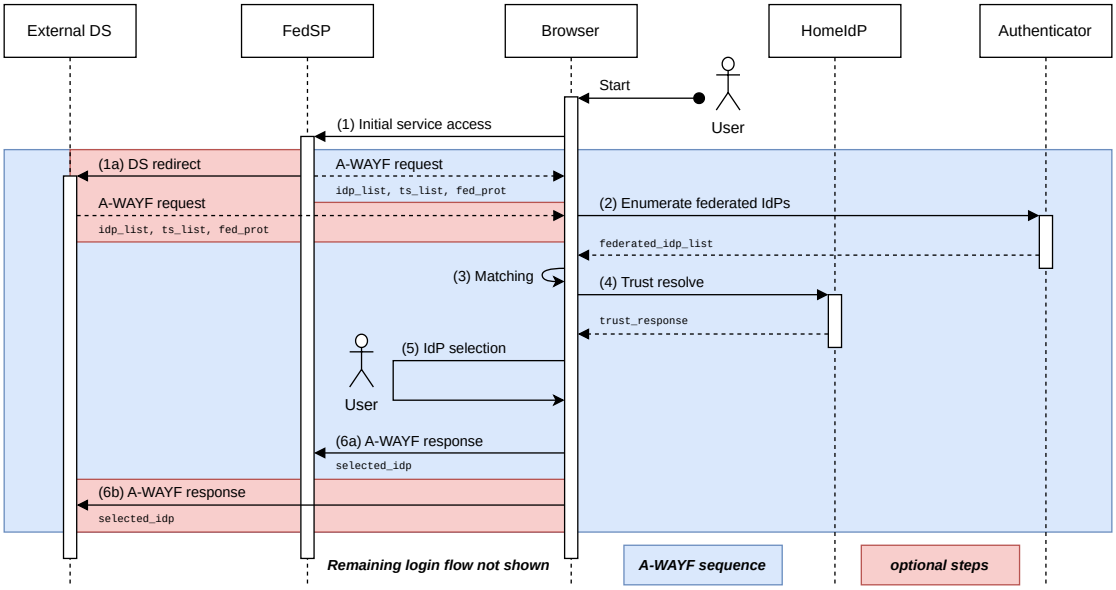

Figure 1: Generic A-WAYF protocol flow as part of federated service access. Steps specific to A-WAYF are shown in blue, whereas optional steps that use an external discovery service are highlighted in red.

Nonetheless, their proposal to include a federation metadata service in the browser [18] could allow A-WAYF to shortcut its trust-resolving process. FedCM focuses on making the subsequent authentication flow more user-friendly without affecting privacy. Therefore, A-WAYF and FedCM can complement each other, offering users a seamless discovery and authentication experience in multilateral federations.

4. A-WAYF

To automate the WAYF process, our solution uses trusted information stored alongside the user's credential during registration with an IdP. We argue that this approach solves the issue that current WAYF implementations face concerning the impending deprecation of third-party cookies. Preserving user privacy was our primary requirement. Therefore, the user must consent to any authentication process being performed. Additionally, the user's affiliation should not be disclosed to an SP that is not trusted by the user's IdP. As a result, we devised a trust resolution process mediated by a browser that uses pre-existing trust relationships within multilateral federations.

The A-WAYF protocol flow, independent of specific technologies, is depicted in Figure 1. Note that the creation of appropriate credentials, i.e., including federation information, is not depicted. Thus, we assume that these credentials were issued beforehand. Each step in the sequence diagram is explained in detail in the remainder of this section. The user's browser serves as a mediator between the federated SP (FedSP), the DS, which can optionally be external, the user's authenticator, and the IdP of the user's home organization (HomeIdP). To avoid ambiguity regarding terminology, we use the terms "IdP" and "SP" as defined by SAML 2.0.

Initial Service Access (1): The user navigates their browser to the FedSP where the A-WAYF process is initiated. In the first step, the browser is supplied with the list of supported IdPs (`idp_list`), a set of trust statements (TS) specific to the FedSP (`ts_list`), and the federation protocol (`fed_prot`), which defines the format of the TS. TS refers to information that the browser can use to verify the existence of a trust relationship between the user's IdP and the FedSP. This first step can either be performed by the FedSP itself, through an embedded DS, or an external DS in Step (1a), which requires an additional redirect. Regardless of the manner in which the DS is operated, the subsequent trust resolution in Step (4) ensures that the FedSP and the DS are trustworthy.

IdP Enumeration (2): The browser collects a second list of URLs representing IdPs for which the user has credentials. This list is aggregated from trustworthy credential sources, such as hardware authenticators or password managers integrated into the browser. In this step, only credentials accompanied by trusted information are considered.

IdP Matching (3): The browser now decides whether any of the user's credentials originate from an IdP included in the set of allowed IdPs provided by the FedSP. This is achieved by computing the intersection of both lists, resulting in a list of candidate IdPs.

Trust Resolve (4): To prevent potential misuse where malicious SPs attempt to deceive users into revealing their affiliation, the browser must determine whether the user's HomeIdP and the FedSP are members of the same federation. Therefore, the list of candidate IdPs and the set of TSs previously supplied to the browser by the FedSP are used to verify this trust relationship. The following process is executed for each candidate IdP in the list: The browser selects and removes the next TS from the set. If the set is empty, the process is aborted. The browser then validates the selected TS, for example, by querying the candidate IdP. If the validation is not successful, the process starts over. Otherwise, the process ends successfully, determining that the FedSP is trusted by the candidate IdP. This step results in a filtered list of trusted candidate IdPs.

User Dialog (5): To proceed with the A-WAYF process, user consent must be obtained through a mediation dialog. The browser may use well-known API endpoints of the candidate IdPs to request additional information for enriching the UX, such as the organization's logo, a localized name, and other metadata. The dialog and subsequent steps vary based on the number of remaining candidate IdPs: If no candidate is found, the process aborts and a manual fallback IdP-selection process is initiated. In the case of a single candidate, the user is prompted to confirm the use of this organization. If multiple candidates are found, the user is prompted to select the organization to be used.

WAYF Response (6): Finally, the browser transmits the HomeIdP selected by the user to either the FedSP in Step (6a) directly or to the federated DS, if one was used, see Step (6b). The trust resolution conducted in Step (4) ensures that the FedSP is trustworthy. Any type of DS, even if it is operated externally, is therefore trustworthy because of the transitive relation. From here on, the usual federated login process continues, i.e., the user is redirected to the selected HomeIdP and authenticates via existing credentials.

5. Proof of Concept

We implemented a PoC to demonstrate that the generic A-WAYF process described in the previous section can be realized in practice. We built our PoC using FIDO2 because it offers a phishing-resistant and convenient alternative to passwords. For this purpose, we selected the Solo Hacker USB-A (v1.2) key from Solokeys as a FIDO2 authenticator because of its open-source nature and the ability to modify the firmware. OpenID Federation is used as the federation protocol in our PoC. Although the protocol is still under development, it fits our R&E use-case and offers all the required functionality.

As already described, the browser mediates the A-WAYF process. Developing a browser plugin for A-WAYF was not practical because of the required changes in CTAP2, which are infeasible to realize inside the sandboxed browser environment. Therefore, we developed a custom client application that implements the A-WAYF process. Modifying the browser itself would have been a highly complex task that offers little benefit compared with our solution.

We set up a testbed based on the SPID/CIE OIDC Federation SDK [19] including a FedSP, an IdP, and a trust anchor. In our testing, the A-WAYF process is directly triggered by our client instead of the FedSP or an integrated DS.

To facilitate the replication of our PoC and testing procedure, we have published the following artifacts on GitHub [20]: The modified firmware for the hardware authenticator along with a virtual authenticator implementation, the client implementation, the full testing protocol with specific testing data, and the configuration of the testbed setup. It was necessary to extend some of the protocols we utilized in our PoC. The following sections describe these extensions in detail. We also explain at which stage of the A-WAYF process they are used. Finally, we delve into the OpenID Federation-based trust resolution process in our PoC.

5.1. WebAuthn (W3C)

We extend WebAuthn with a new extension called `federationId` that has a single property named `idpId`. If this extension is requested by an IdP during a `create()` call, the newly created discoverable credential will use the `idpId` property to store the full URL of the requesting IdP. Apart from supporting the extension when creating credentials, no changes are required in WebAuthn. In our PoC, the creation of the passkey was performed manually.

5.2. Credential Management API (W3C)

We extend the `Navigator.credentials` interface with a new function: `resolveWAYF(...) -> string`. It takes a list of allowed IdP URLs (string), a set of trust statements (opaque), and a federation protocol identifier (string) as arguments. It returns one element of the first list, i.e., the URL of the IdP selected. The format of the trust statements depends on the federation protocol. In the case of OpenID Federation, we propose the trust statements parameter to contain a set of trust chains. This JavaScript function is used by SPs to initiate the A-WAYF process, see Step (1) in Figure 1. Our PoC primarily encompasses the implementation of this function. Instead of being triggered by an SP, our PoC executes it directly.

5.3. CTAP2 (FIDO)

Similar to WebAuthn, we introduce an extension called `federationId` with a property called `idpId`. Furthermore, CTAP2 is extended by the `authenticatorFederationManagement` command. It allows the enumeration of all passkeys that use the `federationId` extension and returns the contained `idpIds` as a set. We propose that this command enforces user verification (UV) every time. Consequently, A-WAYF only works with authenticators that support some form of UV. The command is used by our client, see Step (2) in Figure 1.

5.4. OpenID Federation (OpenID)

In a strict sense, no modifications to the OpenID Federation draft are necessary for A-WAYF. The trust resolution process relies on the IdP's `resolve` endpoint that the specification designates as optional. This endpoint must return the resolved trust chain, which must also include the entity configuration of the trust anchor as the last element in that chain. Moreover, the `resolve` endpoint could be extended by accepting multiple trust anchors instead of a single one. Instead of the browser, the IdP would then iterate over the supplied trust anchors. Consequently, a single request per IdP would be sufficient. We did not implement this change in our PoC because it would offer only marginal performance benefits without affecting the overall concept.

5.5. Trust Resolve

In our PoC, trust resolution occurs in Step (4), see Figure 1. Trust chains starting at the entity in question and ending at a trust anchor can be used to verify federation membership. Therefore, the `ts_list` parameter sent by the FedSP contains one or more lists of JSON Web Tokens representing such trust chains. Our client ensures that each list is internally consistent, i.e., all signatures must be valid. This is possible because all trust chains contain the public keys used for signing. After matching both IdP lists in Step (3), we have a list of candidate IdPs. To determine whether two entities share a common trust anchor in OpenID Federation with minimal browser load, we use the IdP's `resolve` endpoint.

The following process is executed for all candidate IdPs: Our client queries the `.well-known/openid-federation` endpoint of the IdP for determining the `resolve` endpoint. Then, we issue a GET request to that endpoint, including the following: (1) The `sub` parameter, i.e., the `idpId` of the candidate IdP. (2) The `anchor` parameter, i.e., the trust anchor's `entityId`, as stated in the trust chain sent by the FedSP. Using the IdP's own `idpId` as the subject of this request instead of the FedSP's ensures that the IdP cannot infer which SP initiated A-WAYF. If the IdP is unable to resolve a trust chain from itself to the given trust anchor, the SP and the IdP are not part of the same federation. Otherwise, our client validates the consistency of the trust chain returned by the IdP. While both trust chains are internally consistent now, the FedSP could have maliciously generated a trust chain using a forged key for the trust anchor. Therefore, our client verifies that the entity statement of the trust anchor sent by the FedSP was signed using a key contained in the trust anchor's entity statement returned by the IdP. This ensures that both trust chains end with an identical trust anchor. At this stage, we can be sure that the FedSP is trusted by all remaining IdPs.

6. Discussion

We have proposed A-WAYF and demonstrated the feasibility of the concept with the implementation of a PoC. Consequently, several aspects of A-WAYF deserve further discussion. We also describe security considerations and avenues for future work.

6.1. FIDO2 Extension

Instead of leveraging existing data saved alongside the credential, such as the Relying Party ID (RP ID) in the case of WebAuthn credentials, we decided to introduce the `federationId` extension. First, we wanted to differentiate between regular credentials that can be used for one website or domain and federated credentials. Because we introduce a function to enumerate credential information through the FIDO2 client, this differentiation ensures that regular credentials are not affected by this change. Second, a simple flag, e.g., `federated = true`, is insufficient without additional precautions. A-WAYF ensures a one-to-one match between the lists of IdPs from both the FedSP and the authenticator. Consequently, the data format used by these sources must be aligned. However, guaranteeing this alignment would require additional rules for how IdPs set the RP ID. For instance, IdPs may only be reachable via a specific path. Because this information cannot be stored within WebAuthn's RP ID, IdPs would need to offer a separate well-known endpoint for A-WAYF. As we did not want to impose these restrictions on IdPs, our extension stores an additional `idpId`, i.e., the full URL of the IdP. However, if the IdP URL changes, the passkeys will also need to be updated accordingly. We also considered utilizing or extending the existing `authenticatorCredentialManagement` command for A-WAYF. This command exposes advanced authenticator management features that are not supposed to be used in the browser context. Enabling it to be executed by the browser would result in a major change to the overall concept of WebAuthn. Therefore, we did not pursue this idea.

6.2. CTAP2 User Verification

For A-WAYF to gain acceptance, it must be as user-friendly as possible and introduce as little friction as possible. However, it also needs to guarantee that user privacy is preserved by ensuring that information about credentials is never disclosed without consent. To balance both requirements, we propose that the `authenticatorFederationManagement` command can only be executed if UV is enabled on the authenticator and is performed beforehand. This ensures that IdP information cannot be enumerated when the authenticator is lost or stolen. Consequently, if a hardware authenticator is used, the user journey needs to include two dialogs, one for performing UV and the other for choosing the organization. However, we want to stress that this approach can be further optimized from a UX perspective, e.g., by only prompting UV once after power-on or unlocking the device. This suggestion is especially relevant for devices with FIDO2-compatible platform authenticators, such as Windows Hello or Apple TouchID. Here, UV is already performed when the device is unlocked. Therefore, the UV step could be skipped altogether, presumably resulting in a better UX and higher user acceptance of A-WAYF.

Another approach for handling UV requirements in FIDO2 authenticators is the Credential Protection (credProtect) extension for CTAP2 [9]. Its value is set during credential creation and allows the enforcement of policies to protect credentials from unauthenticated access. Therefore, the `authenticatorFederationManagement` command could follow the credProtect policy and apply it to the enumeration of federated IdPs. On the one hand, using the extension and configuring UV to be optional would allow a seamless A-WAYF process without UV. On the other hand, failing to configure the extension would leave the `idpId` vulnerable because credProtect defaults to `optional` protection. Finally, we decided against leveraging the credProtect extension to ensure that user privacy is preserved in any case.

6.3. A-WAYF Sequence

In our solution, the trust relationship between the FedSP and the HomeIdP is resolved before the user selects their organization. Consequently, the time and resources required for the trust resolution process depend on the number of candidate IdPs. If there are many candidate IdPs, this might result in many requests being sent by the browser and an overall slower process. In theory, Steps (4) and (5) in A-WAYF could be switched so that the user's selection is captured before trust between the FedSP and the single selected IdP is resolved. However, the trust resolution process could still fail and result in a greatly disrupted UX. We argue that our sequence of steps is preferable because we do not expect the number of candidate IdPs to be sufficiently high to have a significant impact in most cases. We also find it advantageous to only display organizations with which federated authentication is likely to function properly.

For subsequent visits to a FedSP, cookies can be used to shortcut the A-WAYF process. The user dialog could include an option to skip future dialogs, resulting in a first-party cookie containing the chosen IdP's URL being saved in the browser. The cookie can then be used to automatically redirect the user to the HomeIdP. However, automatic redirection to the HomeIdP is not recommended for DSs [21]. Moreover, the dynamic trust resolution process would be skipped as well, even though trust relationships might have changed since the last visit to the FedSP. Therefore, we decided not to include this functionality in our proposed solution.

6.4. Privacy Considerations

Automating the WAYF process potentially introduces new privacy concerns. Since preserving the user's privacy was the primary requirement, we designed A-WAYF accordingly: First, information about the user's credentials is never disclosed to third parties, even when the authenticator is lost or stolen. Second, the user's affiliation, i.e., the chosen `idpId`, is never disclosed to SPs without an existing trust relationship and the user's consent. Moreover, users might not want to disclose information about the services they visit to one of their IdPs before ensuring that the SP is trusted by the IdP. The A-WAYF protocol cannot guarantee this property in general. In our PoC, this is ensured by transmitting only the SP's trust anchors to the IdP for validation. Finally, our scheme only accesses credential sources to enumerate `idpIds`. In our PoC, a special command is used for that purpose. This ensures that actual credentials are never disclosed.

6.5. Security Considerations

Malicious SPs may send unreasonably long lists of accepted IdPs or trust statements. In this case, the browser can abort the A-WAYF process, falling back to a manual selection. Browsers are therefore required to set sensible limits to prevent these attacks. IdPs may face denial-of-service attacks through the public `resolve` endpoint required by A-WAYF, as requests to this endpoint can impose load on the IdP. However, this attack can be mitigated by restricting public access to the endpoint to requests where the subject matches the IdP's `entityId`. Additionally, both positive and negative responses can be easily cached by the IdP. As detailed in Section 5.4, an extended endpoint in OpenID Federation that adheres to these mitigations may be a viable solution.

6.6. Future Work

The immediate next step for continuing this work is the standardization of our proposed protocol extensions. Beyond that, these extensions need to be adopted by browsers and authenticators.

Our overall scheme is built independently of passkeys. Any password management solution could be used to determine available IdPs in the same manner as passkey-based authenticators. To achieve this, the password management solution would need to store the `idpId` attribute and offer a way for the browser to enumerate them. The same reasoning applies to credentials stored inside the browser itself, and especially to SSI wallets.

Instead of using OpenID Federation, the trust resolution process could also be based on SAML 2.0. Such an implementation would vastly extend the usefulness of A-WAYF because the transition to OpenID Federation is ongoing and many existing federations are still based on SAML 2.0. The absence of a resolve endpoint in SAML 2.0 would necessitate adjustments in the trust resolution process, e.g., a direct validation of metadata in the browser.

7. Conclusion

The process of identifying a user's home identity provider is an ongoing challenge. It is commonly found in multilateral federations, for example, in the research and education sector. Existing solutions either rely on manual selection by the user, resulting in poor UX, or on third-party cookies, which will soon be deprecated. In this paper, we present A-WAYF, a scheme to automate this cumbersome and error-prone process while ensuring maximum user privacy. Our dynamic and browser-mediated process uses trusted information about the IdP stored alongside the user's credentials.

Our solution encompasses two key aspects: Matching the IdPs accepted by a service with the IdPs the user has credential for, and ensuring that a trust relation exists between the service and the user's IdP. The process is generic by design, so future authentication and trust mechanisms can be employed. In addition, we implemented a proof of concept to show that our solution works as designed when using passkeys as credentials and OpenID Federation for trust resolution. Finally, we propose several extensions to existing protocols that enable A-WAYF's functionality, along with a detailed discussion of key aspects.

References

[1] eduGAIN, eduGAIN entities database, 2024. URL: https://technical.edugain.org/entities.

[2] N. Ragouzis, J. Hughes, R. Philpott, E. Maler, P. Madsen, T. Scavo, Security Assertion Markup Language (SAML) V2.0 Technical Overview, Technical Report, OASIS, 2008.

[3] N. Sakimura, J. Bradley, M. Jones, B. De Medeiros, C. Mortimore, OpenID Connect Core 1.0, The OpenID Foundation, 2014.

[4] D. Hardt, The OAuth 2.0 Authorization Framework, RFC 6749, RFC Editor, 2012. URL: http://www.rfc-editor.org/rfc/rfc6749.txt.

[5] R. Hedberg, M. B. Jones, A. A. Solberg, J. Bradley, G. De Marco, V. Dzhuvinov, OpenID Federation 1.0 - draft 32, The OpenID Foundation, 2023.

[6] Coalition for Seamless Access, SeamlessAccess, 2024. URL: https://seamlessaccess.org/.

[7] P. Jones, G. Salgueiro, M. B. Jones, J. Smarr, WebFinger, RFC 7033, 2013. URL: https://www.rfc-editor.orgefo/rfc7033. doi:10.17487/RFC7033.

[8] M. Jones, A. Kumar, E. Lundberg, Web Authentication: An API for accessing Public Key Credentials - Level 3, W3C Working Draft, W3C, 2023. URL: https://www.w3.org/TR/2023/WD-webauthn-3-20230927/.

[9] J. Bradley, J. Hodges, M. B. Jones, A. Kumar, R. Lindemann, J. Verrept, Client to Authenticator Protocol (CTAP), Technical Report, FIDO Alliance, 2023.

[10] Mozilla Foundation, Webauthn browser compatibility, 2023. URL: https://developer.mozilla.org/en-US/docs/Web/API/Web_Authentication_API#browser_compatibility.

[11] OASIS, Profiles for the oasis security assertion markup language (SAML) V2, 2005. URL: https://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf.

[12] T. Kataoka, et al., Leveraging PKI in SAML 2.0 Federation for Enhanced Discovery Service, in: 2009 Ninth Annual International Symposium on Applications and the Internet, 2009, pp. 239–242. doi:10.1109/SAINT.2009.56.

[13] OASIS, Identity Provider Discovery Service protocol and profile, 2008. URL: https://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-idp-discovery.pdf.

[14] SeamlessAccess, Third-party cookie deprecation and its effect on SeamlessAccess, 2023. URL: https://seamlessaccess.org/posts/2023-11-16-3pp-cookies-and-the-sa-button/.

[15] N. Satragno, Credential Management Level 1, W3C Working Draft, W3C, 2024. URL: https://www.w3.org/TR/2024/WD-credential-management-1-20240228/.

[16] N. P. Moreno, Federated Credential Management API, Draft Community Group Report, W3C, 2024. URL: https://fedidcg.github.io/FedCM/.

[17] W3C Federated Identity Community Group, Proposal: IDP-SP-Storage API, GitHub issue, 2023. URL: https://github.com/fedidcg/proposals/issues/4.

[18] W3C Federated Identity Community Group, Proposal: Offloading Trust, GitHub issue, 2023. URL: https://github.com/fedidcg/proposals/issues/5.

[19] G. D. Marco, SPID/CIE OIDC Federation SDK, 2024. URL: https://github.com/italia/spid-cie-oidc-django.

[20] E. Kupris, T. Hilbig, D. P. Sugar, T. Schreck, hm-seclab/paper-a-wayf-spec: A-WAYF support materials, 2024. URL: https://github.com/hm-seclab/paper-a-wayf-spec.

[21] Coalition for Seamless Access, Scenarios - SeamlessAccess Documentation, 2023. URL: https://seamlessaccess.atlassian.net/wiki/spaces/DOCUMENTAT/pages/819234/Scenarios.