Contego
Laboratories

Interoperable, State-approved Electronic Identities

David Basin and Ralf Sasse

January 26, 2018

Contents

Zusammenfassung

Identitätsmanagement bezeichnet Prozesse und Dienste die sich mit der Erfassung, Verwaltung und Verwendung von elektronischen Identitäten (E-ID) befassen. Ein E-ID-Ökosystem kann aus mehreren staatlich anerkannten *Identitätsdienstleistern* (IdP) bestehen. Bei diesen registrieren sich E-ID-*Benutzer* (natürliche Personen) und lassen sich staatlich anerkannte Identifizierungsmittel, wie Passwörter und Security-Tokens, aufsetzen. Benutzer können mittels dieser Identifizierungsmittel ihre Identität und Identitätsattribute, zum Beispiel Name und E-Mail-Adresse, gegenüber Dienstleistern, sogenannten *relying parties* (RP) (auf Deutsch: vertrauenden Beteiligten), sicher authentifizieren. Die Schwierigkeit dieses Ansatzes liegt darin, dass der Benutzer bei einem anderen IdP registriert sein kann, als derjenige mit dem der RP eine Geschäftsbeziehung betreibt und welcher deshalb vom RP zum Prüfen von Identitäten verwendet wird. Nehmen wir zum Beispiel an, dass Sie sich bei Google (Ihrem IdP) angemeldet haben, und Sie eine Dienstleistung (einen RP) in Anspruch nehmen wollen, der Facebook (IdP des RP) verwendet um Benutzer zu authentifizieren. Solch ein Login ist nur möglich falls das verwendete Authentifizierungsprotokoll Interoperabilität unterstützt.

Wir analysieren und vergleichen zwei unterschiedliche Entwurfs-Möglichkeiten für solch staatlich anerkannte E-ID-Systeme mit Interoperabilität. Beide verwenden einen Mechanismus mittels dem der IdP des Benutzers Informationen mit dem IdP des RP austauscht. Die erste Möglichkeit nutzt ein sogenanntes *Federation Hub*. Das Federation Hub ist eine Funktionalität die allen IdP hinzugefügt wird, und die die Weiterleitung von Informationen zwischen dem IdP des RP und dem IdP des Benutzers bewerkstelligt. Die zweite Möglichkeit verwendet eine neue Komponente, genannt *Broker*, bei dem sich alle RP registrieren. Der Broker ist damit der IdP für alle RP und sobald ein Benutzer nun bei einem RP einloggen möchte, kontaktiert der RP den Broker, welcher die Verbindung mit dem IdP des Benutzers herstellt.

Beide Möglichkeiten erreichen Interoperabilität. Zudem haben beide gleichermassen die erwünschte Eigenschaft standardisierte Protokolle verwenden zu können, beispielsweise OpenID Connect, was zur Akzeptanz im Markt beitragen wird. Allerdings hat die erste Möglichkeit einige Vorteile gegenüber der zweiten Möglichkeit. Die erste Möglichkeit ist einfacher, da sie nur eine kleine Protokoll-Erweiterung verwendet, statt der Implementierung einer neuen Komponente wie es in der zweiten Möglichkeit geschieht. Deshalb ist die erste Möglichkeit zudem kostengünstiger. Ausserdem vermeidet sie die Schwachstelle einer zentralen Komponente, die bei Versagen das gesamte System zum Erliegen bringt, wie in der zweiten Möglichkeit vorgesehen. In der ersten Möglichkeit gibt es des weiteren keine einzelne Komponente mit Zugriff auf alle schutzwürdigen privaten Nutzungsdaten aller Benutzer, wohingegen die zweite Möglichkeit eine Komponente beinhaltet, die alle solchen privaten Informationen vorliegen hat, was riskant ist. Allerdings weisen wir darauf hin, dass in Bezug auf Datenschutz und Privatsphärenschutz keine der beiden Möglichkeiten ideal ist, da sowohl in der ersten Möglichkeit die IdPs als auch in der zweiten Möglichkeit der Broker und die IdPs mit Nutzungsdaten der Benutzer betraut werden. Andere Protokolle, zum Beispiel auf Basis von *anonymous credentials*, ermöglichen bessere Eigenschaften in Bezug auf die Privatspäre und minimieren vorliegende Nutzungsdaten, benötigen allerdings eine erhöhte Systemkomplexität.

Summary

Identity management concerns processes and services associated with the collection, management, and use of electronic identities. In an identity-management ecosystem, there can be multiple state-approved *identity providers* (IdPs), where users register and are issued credentials such as passwords or hardware tokens. Users can then use these credentials to authenticate their identity and other personal attributes (e.g., name and email address) to service providers, called *relying parties* (RPs). The challenge in this setting is that the IdP that a user is registered at, may be different than the IdP that an RP has a business relationship with and uses to authenticate users. Imagine, for example, that you have registered just at Google (your IdP) and you wish to use a service (an RP) that uses Facebook (the RP's IdP) to authenticate its users. Logging in would not be possible unless the authentication protocols used support interoperability.

We analyze and compare two different design options for achieving interoperable, state-approved electronic identities. Both implement a mechanism for the user's IdP to exchange information with an RP's IdP. The first option employs a *federation hub*, which is a bit of functionality added to each IdP to support forwarding information between the user's IdP and the RP's IdP. The second option introduces a new component, called a *broker*: every RP registers with the same broker (which is essentially every RP's IdP) and when a user contacts an RP, the broker sets up the connection with the user's IdP.

Both options achieve interoperability. In addition, both share other desirable features such as the ability to leverage existing standards like the OpenID Connect protocol, which will aid market acceptance. Nevertheless, the first option has advantages over the second. It is simpler as it involves just a minor protocol extension rather than the implementation of a new component. It is also less expensive for this reason. Moreover, it avoids a single point of failure as well as a single component that must be trusted to protect sensitive connection information concerning which users are using which services. We caution however that, concerning privacy and data protection, neither option is ideal as the IdPs (in the first option) and broker and IdPs (in the second option) must be trusted to protect connection information. There are alternative protocols, for example using anonymous credentials, that achieve better privacy properties, at the cost of increased complexity.

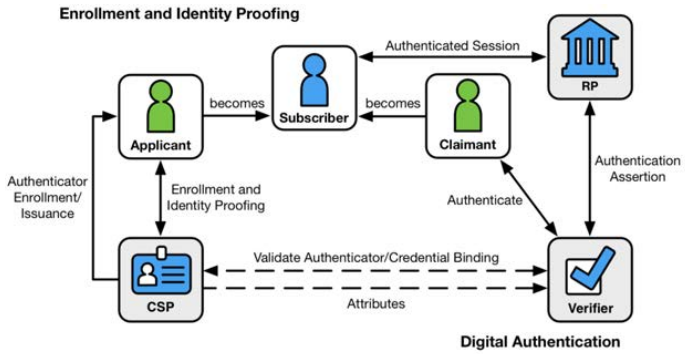

Figure 1: Digital Identity model, graphic from NIST [10].

1 Introduction

1.1 Problem Context

We begin by summarizing the problem of identity management. We use the terminology of the US National Institute of Standards and closely follow [10].

Electronic identities (E-IDs), also known as *digital identities*, are unique digital representations of individuals involved in online transactions. An E-ID is usually associated with one or more *credentials*, also called *authenticators*, such as a user-name/password, a cryptographic token, or a cell phone. An individual uses her authenticators to *authenticate* her identity, for example, to prove that she is who she claims to be when accessing online services.

It is possible for a service provider to maintain its own E-ID registry with associated authenticators. An alternative is to use one or more third parties, called *Identity Providers* (IdP) to handle this task. This is the case when, for example, one uses Google or Facebook as an IdP to log into an online service like an e-shop. In this case, the e-shop, called a *Relying Party* (RP), provides the user the option to login using her Google credentials. The outsourcing to a third party of the collection, storage, and maintenance of E-IDs and the authentication process is known as *Federated Identity Management*.

Figure 1 depicts, at a high-level, the different parties and processes involved in Federated Identity Management. First an applicant must enroll for an E-ID with a *Credential Service Provider* (CSP), which is a trusted entity that registers *subscribers* (also called *users*) and issues or registers their electronic credentials. Upon successful enrollment and identity proofing, the applicant becomes a subscriber who holds an E-ID. A *Verifier* can later check these credentials to authenticate the user and make assertions about the user's electronic identity to RPs, thereby authenticating the user on behalf of the RP. Note that the CSP and Verifier services can be bundled together by an IdP (cf. [9, pg. 36]).

In practice, identity management *protocols* are used to implement some of the arrows in Figure 1. The most widely used protocol of this kind is OpenID Connect [17], or OPIDC for short, which is a suite of protocols built on top of the OAuth 2.0 standard [11]. A representative example of an implementation of OPIDC is *Google Sign-In*, which is supported by many RPs. A user first registers at Google (the IdP), by providing a user name and password, and possibly using a cell-phone as a second authentication factor. Afterwards, the user can click on a *Sign In With Google* button, for example to log into *digitec.ch* (an RP). She is then redirected to Google, where she is asked to log in to her Google account if she is not already logged in. The Google page then displays a message that asks the user to confirm that she wants to log in to *Digitec*. She must further confirm that *Digitec* may access her name and e-mail address. Afterwards, she is redirected to *digitec.ch*, where she is logged in and her user profile already contains her name and e-mail address.

1.2 Problem Statement

The Swiss Federal Council is examining the introduction of a state-approved E-ID and the EJPD has worked out an associated concept that supports multiple state-approved identity providers [9]. A central requirement in [9] is the **interoperability** of the E-IDs. In particular, this requires:

1. The users of a state-approved E-ID can use this E-ID at *all* RPs that accept state-approved E-IDs. Hence, the holder of an E-ID needs a business relationship with just one IdP, and an RP must accept an E-ID, independent of which state-approved IdP issued the E-ID.[1]

2. Each RP needs a contract or business relationship with just one state-approved IdP, as opposed to all such IdPs. Hence, interoperability entails communication between the RP's IdP and any user's IdP.

Note that (2) is important for business reasons and it enables IdPs to bill RPs for services rendered, without requiring contracts between all combinations of RPs and IdPs.

The problem investigated in this report is to compare the different protocol options presented in Section 2 with respect to interoperability and additional technical and economic considerations.

1.3 Resources

For this evaluation, we received the following two documents:

- A description of problem to be analyzed, the different solution scenarios, and the evaluation questions [6].
- A description of the concept of a state-approved E-ID [9].

We also received access to a demonstrator for Option 1b [5], described in Section 2. We additionally used publicly available documents and papers, which are cited in this report.

2 Options Investigated

After introducing several preliminary notions, we present the design options for interoperable federated identity management given in [6]. The first option, consisting of three suboptions, is based on direct interoperability on a protocol basis. The second option is the solution proposed by Identitätsverbund Schweiz (IDV), described in [13].

2.1 Preliminaries

A username is a string of characters (usually without special characters), and the same username can be reused in many systems. A *fully qualified username* is one that additionally identifies the system (or IdP here) where the username is valid. For example, `meier` is a username and `meier@gov.ch` a fully qualified username, where `gov.ch` names the URL of the user `meier`'s IdP.

[1]The E-IDs may have different security levels and RPs may require different levels. We omit these details as they are not relevant for this report.

The OpenID Connect [17] protocol suite defines a *discovery* [18] mechanism that takes as input a URL, usually extracted from a fully qualified username. If a standard-conform IdP is available at this URL, the discovery mechanism extracts the relevant information needed to use the IdP.[2] Discovery often boils down to extracting the URL from a fully qualified username and using the known IdP hosted there.

2.2 Option 1

Option 1 proposes direct interoperability on a protocol basis, e.g., using OpenID Connect. All suboptions may also use OPIDC's *discovery* mechanism. In [6], three possibilities are described:

1a) All IdPs are directly, statically registered with all RPs. This allows the RP to either present the user with a list of all IdPs to choose from, or the user to give her fully qualified username upon which the IdP applies the OPIDC *discovery* mechanism, which then selects the appropriate IdP. The standard OPIDC protocol can be directly used here. See Figure 3 and its description in Appendix A for details.

1b) Each RP is connected with just one IdP and each user is also registered with just one IdP. All IdPs have contracts with each other to exchange billing information. Each IdP also runs a *federation hub* responsible for the discovery of a user's IdP (as in the other suboptions) and relaying information from and to the user's IdP. The federation hub, despite its name, is simply additional functionality built into an IdP that supports the forwarding of user information between two IdPs (the RP's IdP and the user's IdP). In terms of implementation, it amounts to a simple protocol extension on the IdPs' side that extends an IdP to a *federation-hub enabled IdP*. Therefore, when we speak of a "federation hub" we mean just this additional built-in functionality.

This protocol works as follows: A user logging into an RP is sent to the RP-contracted IdP with a list of the requested attributes. The RP's IdP uses its federation hub to discover the user's IdP and to forward the user and attribute list to her IdP. The user's IdP authenticates the user and requests consent to release the attributes listed, and then returns a string of the form `code:User-IdP` to the RP instead of just the OPIDC-usual `code`. Note that the `code` (or `code:User-IdP`) contains no information about the user; it is intended to be redeemed at the issuing IdP who then provides this information. To the RP, this change is opaque, as it just sends this `code:User-IdP` to its IdP as defined in OPIDC. This IdP's federation hub then forwards the `code` to `User-IdP` (the user's IdP) and relays the response token back, which authenticates the user's identifier and attributes for the RP.[3] Option 1b is illustrated with the EJPD demonstrator [5] and is shown in detail in Figure 4 and also described in detail in Appendix A.

1c) This option is a minor variant of 1a. Instead of full static connectivity, the RPs are registered dynamically with IdPs, as needed, as described in [16]. The remainder of the protocol works just like 1a with the caveat that the RP registers at the IdP at runtime, midway through the protocol. See Figure 5 and its description in Appendix A for details.

Only Option 1b satisfies the interoperability requirement described in Section 1.2, in particular with respect to the second part of this requirement stating that an RP only needs a business relationship with one IdP. Hence, we focus on Option 1b exclusively from here on.

[2]In particular, discovery navigates to a specific page, extracts information regarding the supported OPIDC version, cryptographic algorithms used, URL suffixes for different services offered, attributes that are available, modes supported, etc. The RP should revisit this page periodically to check for changes, even when the IdP is known to the RP.

[3]See the Appendix A for further details regarding the secure TLS channel used and signature verification.

2.3 Option 2

Option 2 is the solution presented by [13]. It is described by IDV in [8], which contains the description of the core protocol [8, pg. 17], shown in this report in Figure 6 and described in Appendix A.

Option 2 introduces an additional entity, the IDV *broker*. The broker sits between all RPs and IdPs, so all RPs and IdPs only need a contractual relation with this broker. In the protocol execution, the broker presents itself as an IdP to the RPs, and as an RP to the IdPs. Thus, standard protocol execution can be used on both sides, but is repeated twice per login, once for each side. Additionally, the authentication tokens with attributes generated by the IdP for consumption by the broker need to be translated into the format used by the broker for presentation to the RPs. This is not part of the standard.

3 Interoperability and Tradeoffs

We now address the main evaluation questions raised in [6].

Interoperability

We first examine whether the interoperability of state-approved E-ID systems is achievable with either of the two options described above.

Both options can be used to achieve interoperability in the sense that any user can authenticate to any RP over the IdP she is registered with, independent of the IdP or broker that provides service for the RP. The architectures differ though in how they achieve this. In Option 1b, interoperability is achieved, essentially by forwarding the authentication request and response from the RP's IdP to the user's IdP. In Option 2, interoperability is achieved by using the central broker, whose services are offered by the RP, which enables the user to select his IdP. Hence, both options offer interoperability based on different communication partners and patterns and therefore each option can, in principle, be used as the basis of a Swiss-wide E-ID solution.

Tradeoffs

We next compare the two options' tradeoffs with respect to the metrics given in [6]. We summarize the result of our analysis in Table 2, using a scale of 1-5, where higher numbers are preferable to lower numbers with respect to the given metric, e.g., 1 is worst (or highest cost), and 5 is best (or lowest cost).

Simplicity of implementation: Option 1b requires only minor protocol changes to the standard. Option 2 is significantly harder to implement as the new IDV broker is needed, which is itself both an RP and an IdP with additional token translation functionality. This translation is required, as the RP only knows about the broker and accepts exclusively tokens signed by the broker, while the broker receives tokens signed by IdPs. Thus a translation service must be defined and implemented. Moreover, it must be maintained and upgraded as the standard evolves.

Robustness of operation: Option 1b is relatively robust as a failing IdP only prevents the use of the system by its own customers and not for the customers of other IdPs. In contrast, in Option 2 the IDV broker is a single point of failure. If it fails, then the entire system is unusable. The risks associated with failure can be partially mitigated, in both cases, by using replication to improve robustness.

Topic	Opt. 1b	Opt. 2
Simplicity of implementation	4	2
Robustness of operation	4	2
Usability for E-ID holder	4	4
Usability for RP	5	5
Maintainability (protocol changes)	4	3
Costs	4	2
Market acceptance	5	5
Security	4	3
Data protection	2	1
Overall evaluation	4.0	3.0

Figure 2: Tradeoff table (higher numbers are preferable)

Usability for the E-ID holder: This crucially depends on the number of IdPs in the system. Option 1b's default is for the user to type a fully qualified username, which is inconvenient. If the number of IdPs in the system is small, then the federation hub can also present some buttons (e.g., for 3–5 IdPs), a short drop-down list (for 5–10 IdPs), or a large, searchable list (for over 10 IdPs). For Option 2, usability depends on the exact implementation, but as all IdP are known to the IDV broker, it can work just like Option 1b.

Usability for the RP: In Option 1b, each RP only needs to know about, and implement a connection to, its contractual partner IdP. In Option 2, the partner for the RP is the IDV broker instead. Hence, both are simple to use.

Maintainability with respect to protocol changes: Option 1b requires a small protocol addition, namely the federation hub functionality, which provides discovery and encodes a URI in an otherwise unrestricted code value. If the protocol standard changes the allowed value range for the code, this encoding would need to be adjusted. In contrast, Option 2 possibly requires a change of the translation mechanism, which is completely outside the standard.

Costs: Option 1b requires just one small protocol change and it can otherwise use any of the many existing libraries for OPIDC. In contrast, Option 2 requires implementing and running a new broker service that provides an IdP, an RP, and translation functionality.

Market acceptance: We anticipate market acceptance across the board for both options. For users, one gains the advantages of single sign on in both cases. For RPs and IdPs, one leverages standard protocols. In particular, the RPs have the guarantee that user attributes are authentic (not simply self claimed) and they can use these attributes for their business needs, e.g., partially filling out forms.

Security: Option 1b follows the standard, and if the transmitted location in the code is incorrect or there is a failure, no access would be granted, which is a secure default. Option 2 adds extra complexity due to translation issues, particularly as a mistake in the translation might create incorrect identity tokens that are accepted by all RPs. Therefore, Option 1b is more secure.

We stress that security is a complex issue. It depends not only on the design, but also on the actual implementation and the continued maintenance of the system. Relevant adversary models (e.g., what system components might be compromised) need to be considered in detail for any concrete statement on this point.

Data protection: In Option 1b, all IdPs must be trusted by their customers. However, if one IdP is compromised, the impact of the compromise is at least limited to the IdP's own customers. For Option 2, the IDV broker is not only a single point of failure, it is also a "single point of trust" as it sees all transactions. In both cases, it is therefore essential that the systems and their operations satisfy stringent assurance requirements as the protocol itself does not offer much protection.

Data protection for federated identity management is an important concern as currently the IdP (and/or broker) learns sensitive information about its users. For example, it may learn that a particular user (repeatedly) visits, e.g., a medical forum for patients with an incurable disease, a chatroom for abuse victims, or a help site for people with substance abuse disorders. Revealing this information can have life-changing consequences for the individuals affected.

Overall, the point of a federated identity management protocol suite is to allow RPs to *securely* authenticate users and their attributes. An insecure system is in nobody's interest and will quickly be rejected. Other factors are less critical and different parties may weight them differently. For E-ID holders, usability, data protection, and to some extent robustness are important. For RP and IdP providers, simplicity, maintainability, market acceptance, and cost factors come into play. **Hence we have weighted security 5 times higher than the other factors.** Our overall evaluation, given in Table 2, is the weighted average of these scores.

4 Additional Questions

Question 1: *Can the OPIDC Protocol suite be used to achieve interoperability for the options sketched in Section 2?*

Yes for both options. Option 1b, based on using OPIDC with the minor addition of the federation hub, offers interoperability. OPIDC with *discovery* is used to find the user's IdP, to which the user is forwarded for authentication. To deliver the signed user identifier and attributes back to the RP, the federation hub functionality forwards the redemption *code* from the RP's IdP to the user's IdP, encoded in the `User-IdP` inside `code:User-IdP` and then returns the resulting signed token.

Option 2 as described in [8] uses SAML, but OPIDC could be used instead. As discussed in Section 2.3, implementing Option 2 requires translating an IdP's signed statement about the user's identity to the IDV broker's signed statement about the user's identity. This translation is required as the RP does not have a relationship with the IdP; the RP can only check the signature of the broker, not the IdP, and the IdP's token is marked to be consumed by the broker, rather than for consumption by the RP.

Question 2: *What is the current and expected market penetration of the OPIDC Protocol suite?*

We do not have statistics on the OPIDC protocol suite's market penetration and growth rate. However, it is clear that the protocol suite has gained acceptance and is widely used by numerous major players, both IdPs and RPs. On the IdP side, OPIDC is used by companies like *Google*, *Microsoft*, and *Softbank*, with deployments underway at others, e.g., *Deutsche Telekom.* It is also used by governments such as Chile. On the RP side, its use is also widespread, for example *digitec.ch*, *yelp.ch*, *airbnb.com*, to name a few. Hence the OPIDC ecosystem exists and is thriving.

Question 3: *Is the extension of the OPIDC Protocol suite with names for IdPs (e.g., via URIs) compatible with the suite's existing specification?*

Using the standard *discovery* method shows that no extension is needed to discover the user's IdP, since it can be extracted from the user's fully qualified username. Agreeing on a particular way to encode user names with URIs is not difficult, but care must be taken not to break standard discovery. Encoding the IdP's URI into the return *code* for later redemption, as in `code:User-IdP`, is not forbidden by the current OPIDC specification. Future changes in the specification may possibly require revisiting this. The separator '`:`' is just used as an example and other separators could be used as well.

To summarize, the extension with names for IdPs is compatible with the specification.

Question 4: *Can the OPIDC protocol be integrated into IT Solutions for both IdPs and RPs with reasonable effort?*

Many certified libraries are available for both the RP side and the IdP side, see [14]. Many RPs already support OPIDC and it is widely used (see answer to Question 2). Hence, it is clear that these libraries can be successfully integrated and used.

Question 5: *How good is the OPIDC protocol suite compared with other widely used protocol suites (such as SAML 2.0) concerning data protection and security?*

This question is subtle as it depends on the precise properties desired and the adversary model considered. Also, a secure design does not automatically result in a secure implementation.

Regarding the security of the design itself, there have been attacks found in the past on both SAML 2.0 [1, 19] and OPIDC [7]. This is not surprising as the design of security protocols is difficult and error prone. The attacks on OPIDC were found, and fixed, as part of formal verification attempts for OPIDC. This does not mean that either of the two designs is now without further errors, but just that both have been scrutinized and are at a comparable stand.

With respect to data protection, both are weak. They each give a tremendous amount of information about the customers to IdPs and hence one must trust that the IdPs do not abuse this information or are compromised by an attacker who will abuse this information. This risk can, in part, be mitigated by requiring IdPs to have a high level of security and employing measures suitable for high-assurance systems, including audit and certification.

Note that there are alternative protocols that provide substantially more data protection built-in, namely protocols using *anonymous credentials* [2]. These protocols work without anyone learning at which RP the user logs in, which attributes she shares, and makes her different logins unlinkable. Such protocols [3] are based on primitives like circular encryption [4], however, and therefore are more complex in terms of their design and the cryptography used. While there are examples of their successful deployment in Microsoft's U-Prove [15] and IBM's Identity Mixer [12], they have not yet achieved wide, mainstream use. Moreover, some work may be required to ensure that these protocols satisfy other requirements, e.g., the ability for IdPs to bill RPs for their services.

Question 6: *Can the interoperability between the E-ID systems be realized in other ways and, if so, with what consequences (concerning costs, business risks, etc.)?*

Cryptographic protocols are distributed programs that use cryptography to enable parties to achieve their goals in an adversarial environment. The design space for such protocols is extremely large in terms of how cryptography is used, the communication topology (who exchanges messages with whom), the setup assumptions (e.g., how keys are initially exchanged and whether a PKI is used), how data is serialized (e.g., using data formats like XML or JSON), the properties achieved, the kinds of adversaries the protocol resists, and the criteria discussed in Section 3.

Protocol suites like OPIDC, SAML 2.0, and WS-Security are just examples of options in this design space.

Protocols are typically designed by standardization committees and experience shows it is very difficult to achieve agreement between different industry players. Hence it is advantageous to build on already accepted standards where possible since, in such cases, some agreement has already taken place. Moreover, the standards are often supported by mature libraries, so the costs and business risks are low, compared to starting from scratch with a new protocol design and implementation.

Question 7: *Based on the OPIDC Protocol suite, extended with the convention concerning the identifiers of the E-ID and the RP, can one realize an interoperable E-ID ecosystem, without requiring that each RP has a contract with each IdP (roaming solution with billing for usage)?*

Yes, this can be done using the OPIDC protocol as described in Option 1b. The contracted IdP tracks and bills all requests to the RP, and then settles requests serviced by other IdPs with those IdPs in a pay-per-use manner. All IdPs must have contracts with each other.

Question 8: *Any general advice, reservations, or recommendations?*

As previously explained, with the minor extension of a federation hub, OPIDC can be directly used in an interoperable manner. Our comparison in Section 3 indicates that this is simpler, and, for the most part, advantageous over the use of the IDV broker in Option 2.

Our main concern though are risks to users' privacy, which is a problem with *both* solutions. Short term, these risks can be partially mitigated by ensuring that the systems and associated processes satisfy requirements suitable for high-assurance systems, e.g., comply to standards for federal systems that collect, process, and store sensitive data. In the midterm and long term, protocols should be considered that reduce the need to trust the security of the IdPs or broker, for example, by using anonymous credentials or other mechanisms.

References

[1] Alessandro Armando, Roberto Carbone, Luca Compagna, Jorge Cuéllar, and M. Llanos Tobarra. Formal analysis of SAML 2.0 web browser single sign-on: breaking the saml-based single sign-on for google apps. In Vitaly Shmatikov, editor, *Proceedings of the 6th ACM Workshop on Formal Methods in Security Engineering, FMSE 2008, Alexandria, VA, USA, October 27, 2008*, pages 1–10. ACM, 2008.

[2] J. Camenisch, A. Lehmann, and G. Neven. Electronic identities need private credentials. *IEEE Security Privacy*, 10(1):80–83, Jan 2012.

[3] Jan Camenisch, Maria Dubovitskaya, Robert R. Enderlein, Anja Lehmann, Gregory Neven, Christian Paquin, and Franz-Stefan Preiss. Concepts and languages for privacy-preserving attribute-based authentication. *J. Inf. Sec. Appl.*, 19(1):25–44, 2014.

[4] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, 2001.

[5] EJPD. E-ID-Demonstrator. `http://www.demo-e-id.ch/`.

[6] FedPol. Staatlich anerkannte E-ID — Auftrag "Proof of Concept Interoperabilität E-ID". Project description dated 23.11.2017.

[7] Daniel Fett, Ralf Küsters, and Guido Schmitz. The web SSO standard openid connect: In-depth formal security analysis and security guidelines. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 189–202. IEEE Computer Society, 2017.

[8] AdNovum Informatik AG für Identitätsverbund Schweiz IDV. Interface specification. `https://www.idv-fsi.ch/app/download/10997760494/IDV_24001_Interface_Specification_v0_9_20170831.pdf?t=1505808696`, downloaded 13.01.2018, dated 30.08.2017, version 0.9.

[9] Bundesamt für Polizei fedpol. Staatlich anerkannte elektronische identifizierungsmittel (e-id), konzept 2016. Version from 02.02.2017.

[10] Paul Grassi, Michael Garcia, and James Fenton. Digital identity guidelines. *NIST Special Publication*, 800-63-3, June 2017.

[11] D. Hardt. The OAuth 2.0 authorization framework. RFC 6749, RFC Editor, October 2012. `http://www.rfc-editor.org/rfc/rfc6749.txt`.

[12] IBM. Identity mixer. `https://idemix.wordpress.com/`.

[13] Identitätsverbund Schweiz IDV. IDV website. `https://www.idv-fsi.ch`.

[14] OpenID. Certified openid connect implementations. `http://openid.net/developers/certified/`.

[15] Microsoft Research. U-prove. `https://www.microsoft.com/en-us/research/project/u-prove/`.

[16] Nat Sakimura, John Bradley, and Mike Jones. OpenID Connect Dynamic Client Registration 1.0 incorporating errata set 1, 2014. `https://openid.net/specs/openid-connect-registration-1_0.html`.

[17] Nat Sakimura, John Bradley, Mike Jones, Breno de Medeiros, and Chuck Mortimore. OpenID Connect Core 1.0 incorporating errata set 1, 2014. `http://openid.net/specs/openid-connect-core-1_0.html`.

[18] Nat Sakimura, John Bradley, Mike Jones, and Edmund Jay. OpenID Connect Discovery 1.0 incorporating errata set 1, 2014. `https://openid.net/specs/openid-connect-discovery-1_0.html`.

[19] Juraj Somorovsky, Andreas Mayer, Jörg Schwenk, Marco Kampmann, and Meiko Jensen. On breaking SAML: be whoever you want to be. In Tadayoshi Kohno, editor, *Proceedings of the 21th USENIX Security Symposium, Bellevue, WA, USA, August 8-10, 2012*, pages 397–412. USENIX Association, 2012.

A Appendix: Technical Description of Options

We provide below technical descriptions of Options 1a–1c and Option 2.

A.1 Option 1a

Option 1a is depicted in Figure 3. The user wishes to login at an RP and seven protocol steps are required for this. In Step 1, the user tells the RP her username (this could also be a click on an IdP-identifying button, so that IdP would later ask for the username). In Step 2, the RP extracts the name of the user's IdP using discovery [18] and then forwards the username and the list of attributes needed in Step 3. If a button was used in Step 1, only the list of attributes is sent in Step 3, and the user's IdP must ask the user for her username. In Steps 4 and 5, the IdP asks the user to provide credentials and asks for consent to share the attributes in the list with the RP. If the user successfully authenticates and consents, the IdP sends a signed token with the user identifier and requested attributes to the RP (in implicit mode). Finally, the RP validates the token in Step 7.

Note that there is another protocol mode, called authorization code flow mode. For this mode, in Step 6, the user's IdP would send a code to the RP instead. The RP can subsequently, in a separate step, redeem this code for the identifier and attributes at a web service hosted by the IdP, which only then returns the signed token. This results in 2 additional steps for a total of 9 steps.

A.2 Option 1b

We have worked out the specification for Option 1b based on the EJPD demonstrator [5] to which we had access in January 2018. This option is depicted in Figure 4.

There are four parties involved now, in particular two IdPs. One IdP has a business relation with the RP, called RP-IdP, and the other IdP has a business relation with the user, called User-IdP. Note that if the user and the RP are customers of the same IdP, then only one IdP is involved, simplifying the login procedure. An IdP's federation hub is not mentioned separately in this description as it is an integral component of the IdP itself as explained in Section 2.1.

The protocol now involves 15 steps. In Step 1, the user wishes to login at an RP. The RP sends the user to her IdP in Step 2, including the list of attributes it wants to receive. That IdP requests the user's name in Step 3. Alternatively, if the number of IdPs in the system is small, the IdP could also just provide a set of buttons, or a drop down list, from which the user chooses his IdP. In the general case, the user must enter her fully qualified username in Step 4. In Step 5, the RP-IdP extracts the user's IdP, User-IdP, from the username using the discovery method described in [18]. In Step 6, the username and list of desired attributes are then sent to the User-IdP. The user's IdP can then check the user's credentials at any quality level desired, and check if the user consents to the attribute list for the RP, as shown in Steps 7 and 8. The user's IdP then provisions a code to be redeemed for the requested signed identifier and attributes and sends this code to the user, represented as `Code:User-IdP` in Step 9. The user forwards this code to the RP in Step 10, which sends it to its IdP for redemption in Step 11. The RP-IdP extracts the target IdP's User-IdP from the code in Step 12, and redeems the code in Step 13 at the User-IdP. Finally, the User-IdP provides the signed token with identifier and attributes to the RP-IdP, which forwards that to the RP in Steps 14 and 15, so the RP learns the identifier and requested attributes of the user.

Note that the token is received by the RP over a secure channel that has been established using TLS (Steps 11–15) shared with its IdP, which in turn uses a secure TLS channel to communicate with the user's IdP. Hence the token is guaranteed to be authentic and the RP may rely on just the TLS channel without additional signature validation according to [17, Section 3.1.3.7, step 6]. To additionally validate the signature on the token, the RP requires the issuing IdP's signature verification key. There are several options here: It either knows this key already, looks it up in a PKI infrastructure, or it can receive the key following the OPIDC protocol by registering with the user's IdP as an RP, but without the need for a business relationship.

A.3 Option 1c

For Option 1c, dynamic registration, we have almost the identical procedure as in Option 1a. The only difference is the additional registration of the RP at the User-IdP, in Step 2b, which is added to Step 2. This is illustrated in Figure 5. Note that here the RP cannot provide a set of buttons for the user to select her IdP from as the RP does not know all possible IdPs.

A.4 Option 2

The diagram for Option 2 is taken from [8, pg. 17]. Note that naming is not consistent in the literature. What is called client in the picture is what we previously called the user. The IdP/AA is the same as the previous User-IdP. Instead of the RP-IdP, the RP will contact the IDV broker.

The message flows are as follows. In Step 1, the user (client) requests a page on the RP. This requires authentication, so the RP responds with an authentication request in Step 2. The user forwards this request to the broker in Step 3. In Step 4, the broker computes the possible IdPs. The authentication request contains the list of attributes, which the broker stores for now. In Step 5, the broker presents the user a list of IdP choices, the user picks one in Step 6, and then in Steps 7 and 8 the user is forwarded to her IdP with that request. In Steps 9 and 10, the user is asked for her credentials, at the requested level of authentication. In Steps 11 and 12, the user is forwarded back to the broker with a response containing the IdP signature on the requested attributes. The broker then queries user consent in Step 13, which is confirmed in Step 14. The broker generates an assertion on the IdP response and in Steps 15 and 16 this assertion is forwarded through the user to the RP. The RP can verify the assertion in Step 17 and grant access to the original request in Step 18.

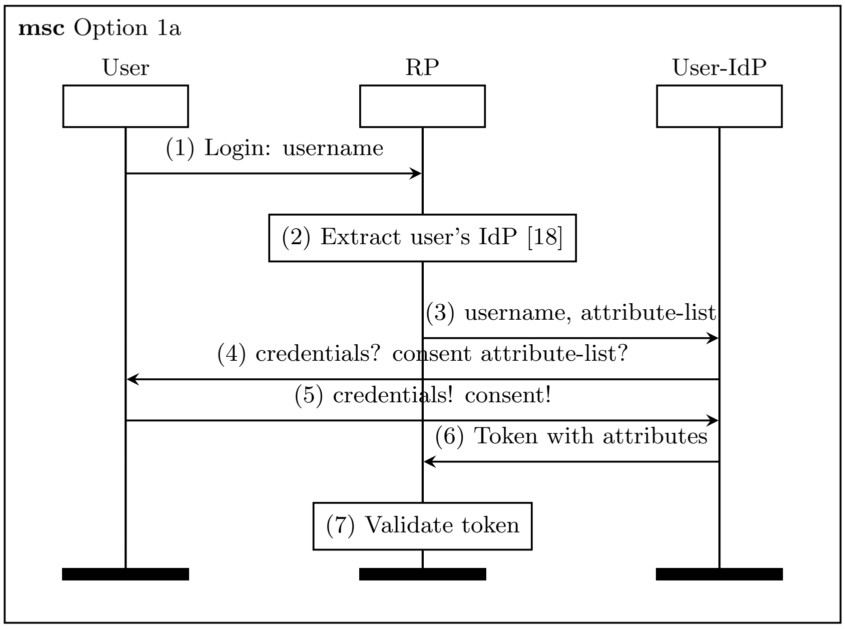
msc Option 1a
User
RP
User-IdP
(1) Login: username
(2) Extract user's IdP [18]
(3) username, attribute-list
(4) credentials? consent attribute-list?
(5) credentials! consent!
(6) Token with attributes
(7) Validate token

Figure 3: Option 1a with full connectivity.

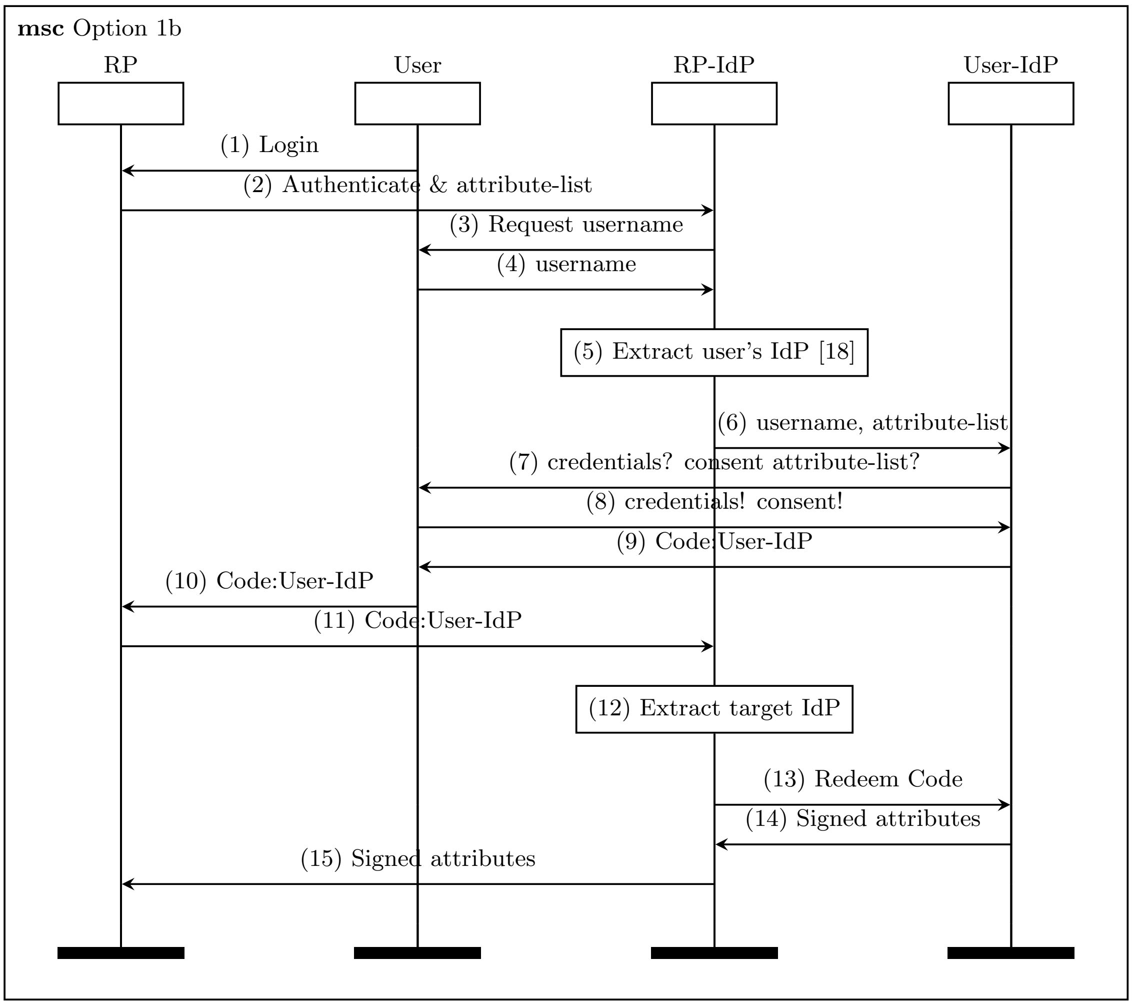

Figure 4: Option 1b with federation hub built into the IdPs.

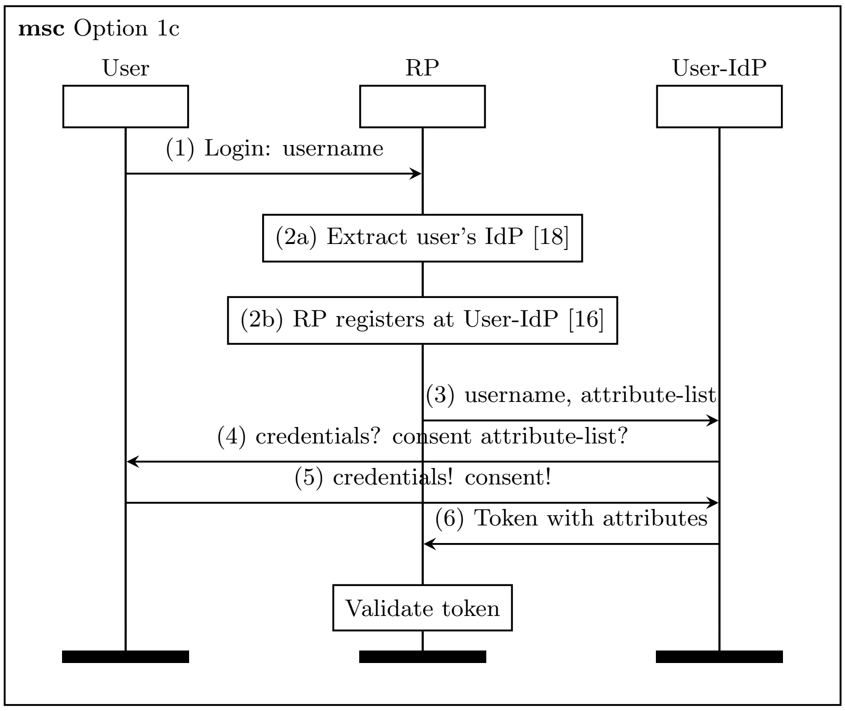
msc Option 1c
User
RP
User-IdP
(1) Login: username
(2a) Extract user's IdP [18]
(2b) RP registers at User-IdP [16]
(3) username, attribute-list
(4) credentials? consent attribute-list?
(5) credentials! consent!
(6) Token with attributes
Validate token

Figure 5: Option 1c with dynamic registration.

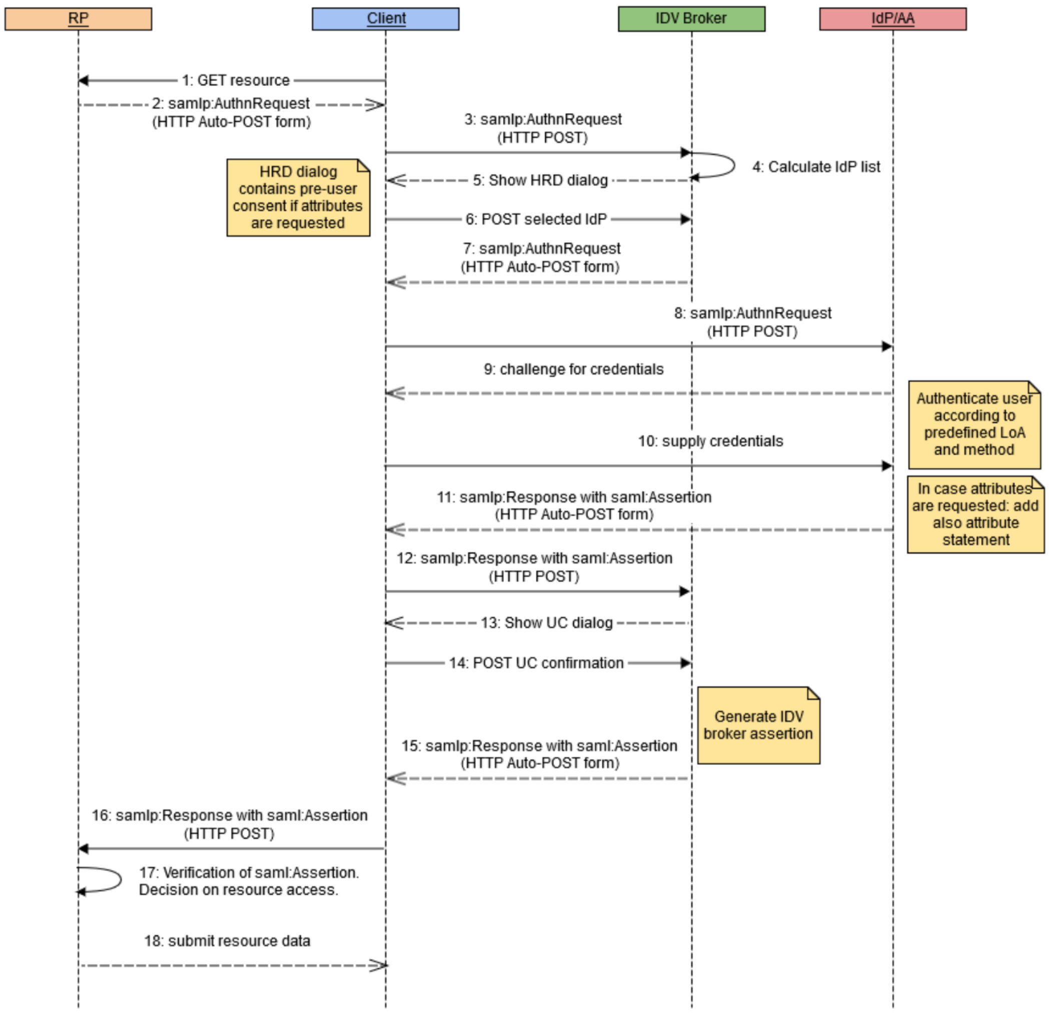
RP
Client
IDV Broker
IdP/AA
1: GET resource
2: samlp:AuthnRequest (HTTP Auto-POST form)
3: samlp:AuthnRequest (HTTP POST)
4: Calculate IdP list
5: Show HRD dialog
HRD dialog contains pre-user consent if attributes are requested
6: POST selected IdP
7: samlp:AuthnRequest (HTTP Auto-POST form)
8: samlp:AuthnRequest (HTTP POST)
9: challenge for credentials
Authenticate user according to predefined LoA and method
10: supply credentials
11: samlp:Response with saml:Assertion (HTTP Auto-POST form)
In case attributes are requested: add also attribute statement
12: samlp:Response with saml:Assertion (HTTP POST)
13: Show UC dialog
14: POST UC confirmation
Generate IDV broker assertion
15: samlp:Response with saml:Assertion (HTTP Auto-POST form)
16: samlp:Response with saml:Assertion (HTTP POST)
17: Verification of saml:Assertion. Decision on resource access.
18: submit resource data

Figure 6: Authentication protocol for Option 2, graphic from page 17 of [8].