

Privacy in Enterprise Identity Federation - Policies for Liberty Single Signon -

Birgit Pfitzmann
IBM Zurich Research Lab
bpf@zurich.ibm.com

Abstract

Cross-domain identity management is gaining significant interest in industry. A recent example is the Liberty Alliance's specifications for single signon of users across a federation of enterprises. These specifications stress that the federation process is voluntary for the users and that privacy is preserved, e.g., by using pseudonyms. We evaluate the privacy of these specifications in detail. We point out ambiguities and propose a concrete privacy policy together with a few changes to the Liberty processing rules. Our analysis demonstrates that identity-management policies are non-trivial even in a limited context. We also discuss how such low-tech proposals from industry relate to high-tech privacy-enhancing proposals from the research community.

1 Introduction

Identity management has many facets. In enterprises, the main emphasis is still on internal consolidation, e.g., on customer-relationship management and on integrating different access channels such as phone and Internet for customers, and Internet and internal systems for employees. In the privacy-research community, the emphasis is on enabling people to manage their identities themselves including free choice of pseudonyms, the transfer of credentials from one pseudonym to another pseudonym of the same person, and appropriate user interfaces. The gap between these facets is wide. Nevertheless, a user-side solution proposed in research can only work if it is taken up on a large scale by enterprises, or if it is compatible on all layers with enterprise-side standards for interacting with users. The latter is quite hard to reach for identity management. (It is easier for mere communication, where some anonymization techniques are transparent to the communication partner.) Furthermore, even though all surveys show that a large majority of the population is concerned about privacy, and about 25% at a considerable price in money or inconvenience, individual users are not good drivers for privacy-enhancing technologies at least given the current ease-of-use and distribution models. This is shown by the results of all companies that tried to commercialize high-end privacy-enhancing technology.

It is therefore essential for the privacy community to also seriously study and try to enhance the privacy achievable by or in interaction with emerging identity-management solutions driven by enterprises. The specifications for single signon across federations of different enterprises recently proposed by the Liberty Alliance may be such an emerging solution, due to the strong membership in this alliance. Liberty is not an open standardization process, but drafts of the second version, 1.1, are available for public comments.

Detailed privacy studies are also important for the enterprises involved in emerging standards because a lack of user trust is a major inhibiting factor for electronic commerce. In other words, a large group of users who are not sufficiently motivated to buy user-side privacy technology are nevertheless sufficiently worried not to use a lot of enterprise-side technology. Specifically for single signon and federated identities, market studies in the wake of Microsoft's Passport product corroborate this clearly. Indeed, the Liberty specifications stress that federating, i.e., choosing single signon between two enterprises, is voluntary for the users and that privacy is preserved, e.g., by using pseudonyms. The minimum goal of such a standard with respect to privacy should be clarity: If users believe in stronger privacy than enterprises do, the users will feel cheated and may start litigation. If enterprises believe in stronger privacy than users do, users will be more reluctant to use the protocols than they need to be.

When looking at Liberty from a privacy perspective, one should be aware that its focus is a business-to-business scenario with small federations with close trust relationships, called circles of trust. (The current protocols do not even scale to large and multiple federations due to assumptions about initial key distribution and specific message formats.) This is different from hosting general-purpose end-user wallets, which is the focus of Microsoft's Passport and of high-tech privacy-enhanced identity management. Liberty's example use case is a federation of airlines and rental-car companies, and a user who already has accounts with two federation members and wants to link them. For instance, bonus points might then accumulate. One could implement a bonus point system nicely with cryptographic credentials, but real airlines and rental-car companies require the user's name and address and relatively strong identification and will not be easily persuaded out of this. Thus most users have to trust these organizations anyway not to exchange undesired information about them, i.e., they have to trust these organizations' explicit or unwritten privacy policies. Of course, there are also business-to-business scenarios where a user wants to be unlinkable even within a small federation or in different interactions with one enterprise.

Studying the privacy provided by a single-signon protocol like Liberty's has two goals: First, make sure that the privacy policies and implications are clearly specified. This is a completely technical goal. Secondly, discuss whether these policies, including the given user options, are suitable for the stated purposes. Indeed we will point out several major and minor privacy-related ambiguities in the Liberty specification. We will propose fixes and an overall policy for the Liberty specifications.

Overview of this Paper

In Section 2, we survey related literature, and in Section 3 we give an overview of the protocol we analyze. In Section 4, we summarize the ambiguities. This is an addition to the introduction, using terminology explained in Section 3. In Section 5, we approach the policy question for Liberty systematically by studying the data that are released in the protocols given certain user choices, and whether this is fully specified or not. In Section 6, we discuss what privacy policies best fit the Liberty protocols. Section 7 gives an outlook and Section 8 a summary.

2 Related Literature

The Liberty specifications are one of several recent specifications of web single signon across different enterprises for users that have nothing but a browser. Accommodating this “zero-footprint” case, at least among others, is currently considered essential for market acceptance. Three parts of the six-part Liberty specification are relevant for us [Libe_02, Libe2_02, Libe4_02].¹ Such browser-based protocols were initiated by Microsoft's Passport product [Mitr_01], which led to many discussions about privacy and points of control. The only technical contributions mainly concerned operational security [KoRu_01, Slem_01]. For a similar product from another company, see [IBM_02] Ch. 10.2. The only open standardization initiative is OASIS's SAML [SAML_02]. Both Liberty and the Internet2-project Shibboleth [Shib_02] are built upon SAML.

We gave an overview of privacy requirements and design consequences for browser-based protocols in [PfWa_02], together with a sketch of a protocol BBAE achieving optimal privacy. (BBAE is available in more detail in [PfWa1_00].) That overview concentrates on general-purpose attribute-exchange protocols, and on design consequences from privacy on message flows and formats. It does not propose specific policies for the choice of names and attributes. The current paper does this for one specific protocol. We chose Liberty for this analysis because it is single-signon only, not very extensible, and even defines some user-interface aspects, i.e., it is really one protocol only, in contrast to the high flexibility of SAML or BBAE. Moreover, it is surprising how complicated a policy becomes even for pure single signon. We guess that the Liberty Alliance postponed attribute-exchange protocols in order to avoid policy issues. The analysis shows that this separation is not possible.

The high end of privacy-enabled identity management is exemplified by the idemix prototype of an anonymous credential system [CaLy_01, CaVa_02]. In particular, this is the first system with efficient multi-show credentials, comprehensive choices as to anonymity revocability, and an all-or-nothing transfer property. Anonymous credentials were first proposed in [Chau_85]. A high-level vision of an overall system built around these ideas and other, complementary privacy techniques like anonymous communication, is given in [CPHV_02]. Anonymous credentials are the only known identity-management solution for cases where a user wants to use unlinkable pseudonyms with different organizations and nevertheless transfer certified attributes between these organizations. However, in current electronic commerce, not many attributes are certified; typically users just fill in forms, and even Microsoft Passport did not certify anything until quite recently, and now only control of email addresses. Hence one can strive for using simple pseudonyms where no certification is needed, simple certificates where no anonymity is possible for other reasons (which may be non-technical and thus changeable), and anonymous credentials in the remaining case.

For a discussion of more remotely related techniques like form fillers and PKIs we refer to the appendix of [PfWa_02]. The pseudonym-choice policies of Liberty resemble [GGKW_99]. However, there the single-signon provider acts as a browser proxy. While browser proxies are a good choice for user-side identity management, they would be very privacy-unfriendly for enterprise identity

¹ Liberty has recently made drafts of V1.1 available (http://www.projectliberty.org/specs/v1_1draft/index.html). All our citations remain to the stable version 1.0, but we verified that no essential changes were made or announced to the parts that we discuss (as of Nov. 29).

management as in Liberty: If one enterprise acts as a user proxy with respect to another enterprise, it sees the user’s entire communication with the second enterprise. In contrast, in all browser-based protocols mentioned above, the single-signon provider only takes part as a server specifically during single signon. Further, this allows the use of multiple single-signon providers per user, e.g., a bank for financial information, a doctor for medical information, and a user-side wallet for personal information. (However, Liberty itself, in contrast to BBAE, does not allow user-side wallets, corresponding to its scenario of small enterprise federations only.)

3 Liberty Single Signon and Federation

We first introduce browser-based single-signon protocols in general, and then special aspects of the Liberty protocol.

3.1 Browser-based Single Signon

The overall structure of all current browser-based single-signon and attribute-exchange protocols is shown in Figure 1.

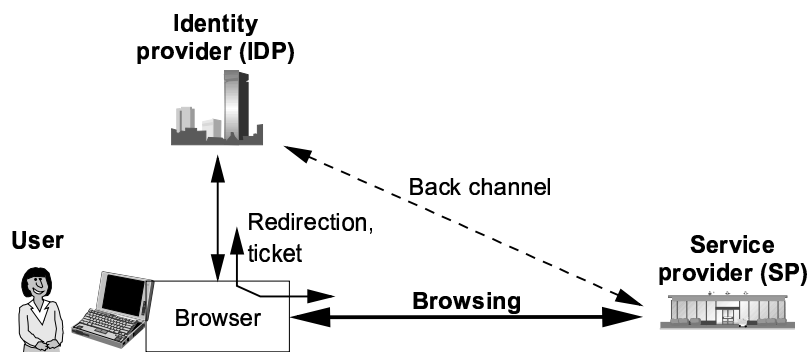


Figure 1 Scenario of browser-based single signon

A user is initially browsing at a service provider. When the user wants to log in (or to send attributes in more general protocols), the service provider redirects the browser to the user’s identity provider. The user logs in there, typically with a fixed user ID and password. The browser and identity provider may also reuse a secure session from another recent login. The identity provider then redirects the browser back to the service provider with some ticket. If the information to be transferred is short, it can be completely included in this ticket. Most protocols also provide a back channel for transferring longer information; the ticket then contains a handle to that information so that the service provider can associate a returning browser can with the appropriate back-channel information.

3.2 Special Aspects of Liberty Federation

The overall Liberty scenario is described in [Libe_02]. A user starts participating by consenting to “federation” of “introductions” at an identity provider. Such a phase is normally called registration; only in Liberty it is assumed that the user already has an account at the identity provider, so that no new data are exchanged. Later, when the user browses at a service provider in the same federation,

the service provider notices that the user has an identity provider in this federation, and asks the user whether he wants to federate or link these two specific accounts (“identities”). The only subprotocol specified for how the service provider notices this is by a cookie in a common federation domain; we assume in the sequel that this subprotocol is used. Then the redirections as in Figure 1 happen for the first time, and the identity provider and service provider exchange a pseudonym by which they will refer to this user. Later signon happens under this pseudonym, again according to Figure 1. The message formats are described in [Libe2_02] and the protocols (“profiles”) in [Libe4_02].

4 Overview of Ambiguities

Before starting the somewhat tedious data analysis, we summarize the main questions that will remain open in Section 4, i.e., for which we will not only extract an existing implicit Liberty policy.

- If a user consents to the federation of two identities that she is using with two organizations, does she consent only to single signon between them, or to arbitrary background information sharing, or does this depend on policies?
- Can a service provider federate, i.e., link accounts, even without user consent?
- How can a service provider restrict single signon to situations after federation?

5 Data Released in Liberty Protocols

We now approach the privacy question systematically by looking at the types of consent in Liberty V1.0 and the technical consequences, in order to identify which data releases the Liberty policies must cover as a minimum.

5.1 Liberty Consents

A user in Liberty has two choices, corresponding to opt-ins to certain data releases:

1. When logged in at an identity provider *IDP*, the user can allow to federate the current identity in principle. We call this “introduction consent”.
2. When interacting with a service provider *SP*, the user can allow to federate her current identity with that at a specific identity provider *IDP* of the same federation. We call this “federation consent”. The use cases suggest that the user must be logged in at *SP*, and thus a priori known at *SP* (in particular [Libe_02], p.10), but technically nothing is based on this.

5.2 Liberty Data Releases

The following data releases happen in the Liberty protocols.

5.2.1 Directly After Introduction Consent

Upon introduction consent, *IDP* sets a cookie on the user’s browser in a federation domain. Recall that we assume that the only specified protocol for this phase is used; it is the “Identity Provider Introduction” from [Libe4_02], Section 3.6. This tells all other members of this domain that the user has identity provider *IDP*.

It is explicitly left open whether the cookie is persistent or session-based, i.e., whether it only divulges the name *IDP* or the fact that the user is currently logged in at *IDP*.

If new members join the federation, they automatically also receive introductions by this technique.

5.2.2 Federation

At any time after introduction consent, *IDP* is willing to accept federation requests from service providers in its federation. A federation request is a single signon request with an element `<Federate>=true` [Libe2_02].

The first problem is whether introduction consent also allows the identity provider to federate with a specific service provider. The Policy/Security Note after Figure 4 in [Libe_02] strongly suggests that it does not: “In Figure 4 the user is not consenting to federating his identity with any service providers. Soliciting consent to identity federation is a separate step, as illustrated in Figure 5.” However, technically it does: Assume user *U* wants to enable federation at an identity provider *IDP* of a federation *F*, but not with federation member *SP* because *U* does not fully trust *SP*. However, the following consent for federating with *SP* is only given to *SP*, and *IDP* simply believes it when obtaining the federation request. Hence if *SP* is indeed untrustworthy, it can get the federation without *U*'s consent. By using the element `<IsPassive>` in the request, *SP* can even ensure that *IDP* does not contact *U* during this request, so that *U* cannot notice this and complain.

Upon a federation request, *IDP* generates a new pseudonym $id_{U,SP}$ for user *U* in interaction with *SP*, and is from then on willing to always authenticate *U* to *SP* under $id_{U,SP}$.² The rules for pseudonym generation are missing in the processing rules for the request, Section 3.2.3 of [Libe2_02], but are defined at the beginning of Section 3.3: “At the time of federation, the identity provider generates an opaque handle that serves as the name identifier the service provider and the identity provider use in referring to the Principal when communicating with each other.”

5.2.3 Starting Single Signon

After federation consent, *SP* makes a federation request to *IDP*, and can then ask *IDP* to authenticate *U* at any future time. The text is quite specific that single signon should only be used after federation ([Libe_02], Sections 2.2 and 5.4.2). However, it is not specified how *SP*, at the moment where it desires single signon, knows whether it has federated for this user: Even if *SP* has an account for *U* and noted the federation there, at this moment *U* has not been authenticated. A possibility is that *SP* has set a persistent cookie on *U*'s browser and desires the single signon only for better security. However, the cookie may be absent because *U* switched cookies off or uses multiple browsers.

- Now either *SP* might simply try single signon, implying certain data releases (see Section 5.2.4) even if *U* did not consent to federation.
- Or *SP* asks for user consent now; we call this “single-signon consent”.

5.2.4 Data in Single Signon

In each single signon *SP* tells *IDP* that *U* is currently browsing there, and gets *U* authenticated under $id_{U,SP}$. More precisely, first *IDP* obtains the information that the user whom *IDP* knows under a local

² *SP* may ask to have this pseudonym replaced by another one, using the name registration protocol, but this makes no difference for privacy, only that *SP* may need one name less internally.

name $id_{U,IDP}$ is currently browsing at SP , while SP itself does not know this. This happens transparently if U is authenticated at IDP at that time; otherwise U gives some additional implicit consent by authenticating to IDP . Then IDP tells SP that this is the user known as $id_{U,SP}$ at SP .

5.2.5 Attribute Exchange

No exchange of further attributes of U , in particular of names or addresses, happens directly by the Liberty protocols. However, the existence of a common pseudonym enables the identity provider and the service provider to exchange data about the user by other protocols. The main question, as anticipated in Section 4, is whether federation consent for two organizations implies

- a) consent only to single signon between the two organizations,
- b) or to arbitrary background information sharing between them,
- c) or whether this, and the extent of the sharing, depends on policies specific to the federation or the current two organizations.

This must be stated clearly, and in Case c), a clear user interface is needed for looking up these policies prior to consenting. To see that this is indeed unclear, compare the following information:

- From the accompanying press release (<http://www.projectliberty.org/press/releases/2002-07-15-1.html>): “The Liberty version 1.0 specifications do not involve the exchange of personal information. Instead, they involve a format for exchanging authentication information between companies so the identity of the user is safe, and specific details about the customer’s identity are not shared.” This sounds like Case a), and most users will understand it like this and expect background information-exchange to be forbidden. However, one can take the position that also in Case b) and c), the *specifications* do not involve what happens in the background.
- The discussion that providers cannot skip over each other in chains of linked identities in [Libe_02], Section 5.4.1, also sounds like Case a), because it suggests that no other attributes like names are exchanged that would be the same throughout the chain.
- The notion of “account linking” in [Libe_02], Section 2, and the general network-identity vision in Section 1.2, sound like Case b).
- The unspecified background web services in [Libe_02], Figure 11, look like Case b), but instead they may only be the federated logout service.
- After Figure 17 in [Libe_02]: “The semantics of such a federated relationship between identity providers are not dictated by the underlying Liberty protocols. These semantics will need to be addressed by the agreements between the identity providers and the capabilities of the deployed Liberty-enabled implementations.” This sounds like Case c), although only for the special case of two identity providers.
- The example screen asking the user for federation consent, Figure 5 in [Libe_02], contains no link for help or for looking up the federation rules. This looks as if it cannot be Case c).

5.3 Withdrawing Consent

No protocol is specified for undoing introduction consent, but it can easily be added by asking the identity provider to delete the cookie. We call this “revoking introduction consent”. As all other interactions can be terminated by the user, we assume that this is also intended in Liberty. Users may

also want to revoke introduction consent for reasons other than privacy, in particular for changing their identity provider.

Federation consent can be undone at either *IDP* or *SP* by “federation termination”, also called “defederation” (Section 5.4.1.2 in [Libe_02] and Section 3.4 in [Libe4_02]). Either party has to notify the other, and from then on, *SP* should not send authentication requests to *IDP*, nor *IDP* answer them from *SP*. If *SP* and *IDP* later federate again, *IDP* generates a new pseudonym as described above. This implies that by federation termination at *IDP*, user *U* can reliably cut the link between its prior interactions with *SP* and future ones (while interacting in two roles with *SP* in parallel is not possible). Of course, if federation occurs both times from an existing account with *SP*, then *SP* can link all interactions with *U* anyway.

6 Policy Proposals

Now we propose concrete policy rules, structured according to the different data categories that we have seen. We sometimes present several options, but we propose to fix one clear policy for usage with the Liberty V1.0 specifications or similar specifications, leaving a larger choice of policies to a future version that also offers a larger choice of techniques. In other words, simplicity is the main advantage of having only single signon, fixed federations, and fixed roles, and this should be reflected by a clear and simple policy.

6.1 Overview Table

Table 1 gives a detailed overview of the data types that we saw and some that might occur in future extensions, together with the main policy options as discussed already and in the rest of Section 6. A short summary of the final recommendation will be given in Section 8.

6.2 Introduction Data

The data that introduce an identity provider are not very sensitive, in particular as long as they do not contain any details such as the responsibilities of a particular identity provider with respect to a user. Releasing them improves user convenience, which seems the main motivation for a pure single-signon protocol. Hence we propose a lax privacy rule:

Rule_{intro}: If user *U* gives introduction consent, *IDP* may tell arbitrary recipients where *U* is browsing that this user has identity provider *IDP*. This holds until future opt-out by revoking introduction consent; an easy interface for this must be provided.

Note that this release is not linked to any name or pseudonym of *U*, just to “the current browser user”. Furthermore, we have allowed release to all future federation members as needed for the Liberty introduction protocol, and even to outsiders, so that the common-domain cookie does not need special protection. (Actually, the Liberty recommendations do provide this stronger protection.) We decided that the introduction data should be the name *IDP*, not the current login status, mainly because it is more convenient for the user if single signon also starts if the user is not logged in at *IDP* yet.

Data Category	Detailed data	Consent needed at	Exists in Liberty SSO?	Realization (Liberty SSO or extensions)	Recommendation for Liberty SSO policy	Recommended side-effects of rule
General						Later opt-out (termination). Minimum dispute resolution and assurance standards.
Introduction	<i>U</i> 's identity provider	<i>IDP</i>	Yes, or 3	Cookie from <i>IDP</i> (or header, script etc.) if persistent	Do after introduction consent	Access anyway by cookie. Retention limit.
	Responsibility of <i>IDP</i> (e.g., financial or work)	<i>IDP</i>	No	Cookie etc. as above	n/a (= not applicable)	n/a
	Login status	<i>IDP</i>	Yes, or 1	Cookie etc. if per session	Don't	n/a (else retention limit)
Traffic	Name of <i>SP</i> and fact that <i>U</i> is now browsing there	<i>SP</i> . For transparent use via <i>IDP</i> ?	Yes	Needed for redirect back and man-in-the-middle security	Do after federation consent or single-signon consent at <i>SP</i>	Implicit access to <i>SP</i> -names, retention limit. (If at <i>IDP</i> , audit and harsh punishment for abuse)
	Anything else about <i>U</i> 's current actions at <i>SP</i>	<i>SP</i>	Maybe	Exact accessed URL; element <code><RelayState></code>	Don't	n/a (else retention limit)
Names	Fixed role name per service provider	<i>IDP</i> . For transparent use via <i>SP</i> ?	Yes	Name in authentication token, or name or attribute in attribute token	Do after federation consent at <i>IDP</i>	Retention limit. (If at <i>SP</i> , audit and punishment for abuse.)
	Name with a-priori meaning	<i>IDP</i>	No	As above	n/a	n/a
	Freely chosen role	<i>IDP</i>	No	As above	n/a	n/a
Other user attributes	Arbitrary	<i>IDP</i> (and <i>SP</i> if bidirectional exchange)	Maybe	Attribute tokens, or background exchange enabled by linking	Consent to policy with federation consent at <i>IDP</i> . Only <i>IDP</i> .	Require a seal for policy. Details in policy.
	What attributes <i>SP</i> wants	<i>SP</i>	Maybe	Attribute queries	Consent to policy with federation consent at <i>SP</i>	Require a seal for policy. Details in policy.

Table 1 Overview of data categories and recommended policy rules for their release.

6.3 Authentication Data

Authentication data are much more critical than introduction data. We recommend the following rule for identity providers:

Rule_{IDP,auth}: Introduction consent is for introduction only, without any effect on authentication. User *U* can separately give federation consent at *IDP* for every specific service provider *SP* to

allow federation with *SP*. Then *IDP* may authenticate *U* under a fresh, but then fixed role pseudonym $id_{U,SP}$ to *SP* whenever *SP* asks, until future opt-out by federation termination.

This rule implies a small change to Liberty’s processing rules (Section 3.2.3 of [Libe2_02]): *IDP* itself must ask for a user OK when federating with a new service provider *SP*. This is fourth type of consent after the original two from Liberty (Section 5.1) and the potential single-signon consent from Section 5.2.3. In future versions with flexible protocols and privacy policies, a user *U* can also pre-authorize this for arbitrary sets of service providers.

The following rule is closer to the current Liberty specifications, but only our second choice:

Rule_{IDP,auth,2nd}: If user *U* gives introduction consent at *IDP*, then *IDP* may authenticate *U* under a fresh, but then fixed role pseudonym $id_{U,SP}$ to any service provider *SP* that was a member of *IDP*’s federation *F* at the time of *U*’s choice. The list of members of *F* must be easily retrievable when *U* makes the choice. This holds until future opt-out by federation termination.

The restriction to federation members at the time of consent in this rule is necessary to keep *U* in control. Otherwise the risk is large that some federation will grow to include almost every company in the world, and suddenly *U* will be authenticated under fixed pseudonyms with service providers where she never intended that. Disadvantages of this rule over our recommendation are:

- Identity providers must store for which federation members *U* gave consent, and will probably want a user interface for consent for new members, so that all technical additions needed for Rule_{IDP,auth} are also needed here.
- In larger federations, almost no user will want federation with all service providers, both for privacy and for convenience (this type of federation does not allow multiple roles).
- Audit and minimum punishments are otherwise needed to make Rule_{SP,auth} credible, i.e., to deter dishonest service providers from federating with *IDP* against the user’s wish.
- It works less well together with the rule we propose for attributes (Section 6.5).

For the service provider, introduction consent has no effect (for both versions of the rule for the identity provider.) In particular, *SP* should not collect introduction data or contact *IDP* for users that do not choose to federate at *SP*.

Rule_{SP,auth}: If user *U* gives federation consent for an identity provider *IDP* at *SP*, then *SP* may record this choice (e.g., by setting cookies on the user’s browser) and the pseudonym $id_{U,SP}$ of this user, and may link different interactions with this user by this pseudonym. If the choice was made from an existing account, it may also link these interactions to the existing account.

6.4 Traffic Data

Traffic data arise at the identity providers because service providers notify the identity provider that the user is browsing there. These data arise implicitly in the protocol and are not needed for the applications that a user expects. Hence we propose strict privacy rules for them.

In particular we require that a service provider who is not sure about federation consent asks the user for single-signon consent, i.e., we opt for the second solution to the problem from Section 5.2.3. Interaction with *U* is typically needed anyway in this situation because the *IDP*-introducer cookie is also absent; such interaction is described in [Libe_02], Section 5.4.3.5. An advantage of this solution is that the alternative would need strict audit whether some identity providers introduce themselves

for users that never gave introduction consent there, in order to collect visited-sites trails from service providers in this situation.

Rule_{IDP,traffic}: (Given introduction consent.) An identity provider *IDP* must not mine traffic data or use them for any other purpose than single signon. He must not forward them to any other party. Exceptions may only be given by law (e.g., storage requirements for law enforcement) and for authentication classes where dispute resolution is offered (i.e., where a service provider and a user may need records from *IDP* about an authentication that the user denies).

Rule_{SP,traffic}: (Given federation consent or single-signon consent.) A service provider *SP* must only provide fixed data about itself to the identity provider in single signon and federation, not user-dependent data.

Consequently, *SP* should not put unencrypted user data in the element `<RelayState>` of a single-signon request, in particular not the exact URL that the user wanted to access. The contradicting recommendation in Section 3.2.1 of [Libe4_02] should be modified (in Step 3, recommending to use the element `LRURL=<return URL>` from Step 1.) Random values and data encrypted with a key known only to *SP* are permitted.

6.5 User Attributes

Now we come to the question of user attributes. As they do not occur explicitly in the Liberty V1.0 specifications, we are technically free to decide about them. We will recommend a certain version of Case c) from Section 5.2.5, i.e., identity-provider-specific policies.

From a privacy perspective it is tempting to recommend Case a) instead, i.e., essentially no attribute sharing; hence we discuss this first. The rule could look as follows:

Rule_{attributes,(a)}: No consent implies any permissions beyond the policy rules from Sections 6.2 to 6.4. In particular, an identity provider or service provider must not share any data about a user they both know as $id_{U,SP}$ using this pseudonym, beyond what is allowed in these rules, nor must they use the common domain for any cookies beyond the specified introduction cookies.

The term “using this pseudonym” already weakens the rule by allowing other forms of sharing. This is unavoidable in particular in closed federations, because some federation members will already be sharing information about these users, e.g., employers and travel agents about travelling employees.

While this rule is reasonable in itself, we cannot imagine that typical federations offer single signon with so little benefit to themselves. The other extreme, an implicit permission to share arbitrary data (Case b), is out of the question because it contradicts all fair information practices. We therefore assume that the Liberty Alliance meant some form of Case c). This, however, requires a reference to a privacy policy. It seems impossible to propose just one policy (even with open parts for user choices) for all federations. Hence we recommend the following rules:

Rule_{IDP,attributes,(c)}: If user *U* consents at *IDP* to federate with a service provider *SP*, then *IDP* may use the generated pseudonym $id_{U,SP}$ to provide information about *U* to *SP*, provided *U* also consented to a privacy policy that allows this. The policy must be explicitly referred to and easy to look up in detail before the consent, and withholding consent must be easy. The policy should at least have a seal from a well-known organization. The permission holds until future opt-out by defederation. To what extent sent attributes may survive defederation must be clarified in the policy, as well as to what extent *IDP* may store a history of *SP*'s requests.

Rule_{SP,attributes,(c)}: If user U consents at SP to federate with an identity provider IDP , then SP may use the provided pseudonym $id_{U,SP}$ to ask IDP for attributes about U and to use the obtained attributes, provided U also consented to a privacy policy that allows this. The policy must be explicitly referred to and easy to look up in detail before the consent, and withholding consent must be easy. The policy should at least have a seal from a well-known organization. The permission holds until future opt-out by defederation. To what extent received attributes survive defederation must be clarified in the policy.

Rule_{cookies}: The common domain is not to be used for cookies except the specified introduction cookies.

The following points should be noted about these rules:

- The rules are asymmetric, i.e., they assume that attributes are only transferred from IDP to SP . This is more user-friendly given that a distinction between identity providers and service providers is made. For bidirectional exchange, the organization SP should also act as an identity provider towards the organization IDP , i.e., get separate user consent for sharing its own data.
- As in Rule_{attributes,(a)}, the organizations may still have separate legacy processes for sharing data in ways that may no longer even be known and that therefore have no privacy policy. However, all processes that use $id_{U,SP}$ are necessarily new and known. Therefore it is reasonable to set up a privacy policy when setting up these processes.

6.6 Further Data?

We believe that we have covered all data occurring in the context of the Liberty V1.0 specifications, except for traffic analysis possible in networks. This is not made significantly easier by these protocols if all data are sent over secure channels, and thus we do not consider it further. Extensions to the protocols to larger or more dynamic federations could include key distribution centers or other central directories; if this is done in a way that enables the centers to collect usage trails, the policies put up for consent must govern these data.

6.7 Further Policy Aspects

So far, we have only considered rules for the data releases from one party to another. Privacy policies also govern other aspects, in particular conflict-resolution procedures, user-access rights, notification, and retention periods. In addition to these functional elements, there can be assurance elements such as promises of regular audit, enterprise-internal need-to-know policies, or security evaluations. While we also propose that Liberty fixes as much of this as possible to retain the simplicity of V1.0, we only sketch our recommendations:

- **Termination:** For all consent, the possibility for a later opt-out is already provided.
- **Dispute resolution:** A minimum standard for dispute resolution should be set. We recommend at least a contact address at each IDP and a fixed address per federation as a second resort. Further, while law suits are then hopefully not needed, it seems clear that breach of privacy promises can be a basis for litigation independent of whether this is offered as an explicit dispute resolution type in the policy.

- **Notification:** Under the recommended policy, no specific user notification is needed, because all releases are governed by policies that were consented to. Notification about certain attribute releases can be promised indirectly in the attribute-release policy.
- **Access rights for the user:** Under the recommended policy, no specific user access rights are needed.³ Access rights for certain attribute releases can be promised indirectly in the attribute-release policy. Access to the names of service providers that were federated with an identity provider *IDP* is useful, but must be given anyway in the user interface for defederation at *IDP*.
- **Retention periods:** Each identity and service provider should state retention periods for all data covered by these policies. In addition, a general upper bound on the retention of traffic data seems useful, e.g., one month unless local law or the authentication class require a longer period.
- **Assurance:** Certain minimum assurance standards should be fixed at least for identity providers. Attribute policies at identity providers can additionally require assurance for service providers that receive certain attributes.

7 Outlook

We already sometimes mentioned future extensions to larger and more dynamic federations and to protocols with integrated attribute exchange. The recommended rules should scale well to those cases. We foresee the following most important differences:

- Attribute-exchange protocols can deal better with incomplete policies. This is important for scenarios where an identity provider mainly serves as a trusted user agent following a user-chosen policy, because most people are neither willing nor able to initially set the entire policy of what information they want to share with whom. (In contrast, in some closed federations like supply chains, the users are mainly agents of the federation partners. Then the policy can be fixed by the identity providers.) The attribute-exchange protocol can then contain a real-time release of the attributes.
- Attribute-exchange protocols also enable the provision of demographic or preference data about an anonymous user for whom not even a long-term pseudonym is provided.
- Authentication information will become a special case of attributes, because users may have more than one pseudonym with one service provider, or the same pseudonym with several service providers. In other words, a Liberty V1.0-style pseudonym is just one type of name.
- In the general case, there is no need for a special semantics of “federate” any more, i.e., the user choices do not need to be bundled in the same way. (This may remain one option.)
- In a general e-commerce scenario, users should also be allowed to be their own identity providers, in particular for voluntary attributes like preferences, i.e., to have user-side wallets. Then the names, addresses, and keys of these wallets are personal information and must be covered by the policies. How this reflects into the protocol design was investigated in [PFWa_02].

³ In contrast, policies where a first party believes a second party that a user gave consent at the second party need access rights to enable misuse detection. This would hold for the second-choice rule $\text{Rule}_{IDP,auth,2nd}$ and if $\text{Rule}_{SP,traffic}$ did not have the condition of federation or single-signon consent.

8 Summary

We have analyzed the privacy effects of a web single-signon and identity-federation protocol, specifically the Liberty V1.0 specifications. Although single signon seems quite a fixed notion in contrast to more general attribute exchange, there were a number of privacy ambiguities, and we discussed options for resolving them. We proposed precise recommended policy rules and some alternatives; the recommended policy is summarized in Table 2. We described small changes to Liberty’s processing rules needed to support this policy, in particular two new types of consent.

Data Category	Detailed data	Consent needed at	Exists in Liberty SSO V1.0?	Recommendation for Liberty SSO policy	Recommended side-effects of rule
General					Later termination; minimum dispute resolution and protection standards
Intro-duction	<i>U</i> ’s identity provider	<i>IDP</i>	Yes	Do after introduction consent	Retention limit
	Login status	<i>IDP</i>	Maybe	Don’t	n/a
Traffic	Name of <i>SP</i> and fact that <i>U</i> is now browsing there	<i>SP</i>	Yes	Do after federation or single-signon consent at <i>SP</i>	Retention limit
	Anything else about <i>U</i> ’s current actions at <i>SP</i>	<i>SP</i>	Maybe	Don’t	n/a
Names	Fixed role name per <i>SP</i>	<i>IDP</i>	Yes	Do after federation consent at <i>IDP</i>	Retention limit
Other user at-tributes	Arbitrary	<i>IDP</i>	Maybe	Consent to policy with fe-deration consent at <i>IDP</i>	Require a seal for policy. Details in policy.
	What attributes <i>SP</i> wants	<i>SP</i>	Maybe	Consent to policy with fe-deration consent at <i>SP</i>	Policy mainly covers <i>SP</i> ’s usage of received attributes

Table 2 Summary of recommended policy rules for Liberty V1.0

Acknowledgements

Thanks to Matthias Schunter and Michael Waidner for interesting discussions about these policy recommendations.

References

- Chau_85 David Chaum: Security without Identification: Transaction Systems to make Big Brother Obsolete; Communications of the ACM 28/10 (1985) 1030-1044
- CaLy_01 Jan Camenisch, Anna Lysyanskaya: An efficient system for non-transferable anonymous credentials with optional anonymity revocation; Eurocrypt 2001, LNCS 2045, Springer-Verlag, Berlin, 93-117

- CaVa_02 Jan Camenisch, Els Van Herreweghen: Design and Implementation of the Idemix Anonymous Credential System; 9th ACM Conference on Computer and Communications Security (CCS), 21-30
- CPHV_02 Sebastian Clauß, Andreas Pfitzmann, Marit Hansen, Andreas Pfitzmann: Privacy-Enhancing Identity Management; The IPTS Report (67) 2002, <http://www.jrc.es/pages/iptsreport/vol67/english/IPT2E676.html>
- GGKM_99 Eran Gabber, Phillip B. Gibbons, David M. Kristol, Yossi Matias, Alain Mayer: Consistent, Yet Anonymous, Web Access with LPWA, Communications of the ACM 42/2 (1999) 42-47
- IBM_02 IBM: Enterprise Security Architecture using IBM Tivoli Security Solutions; April 2002, <http://www.redbooks.ibm.com/abstracts/sg246014.html>
- KoRu_00 David P. Kormann, Aviel D. Rubin: Risks of the Passport Single Signon Protocol; Computer Networks 33 (2001) 51-58
- Libe_02 Liberty Alliance Project: Liberty Architecture Overview, Version 1.0, 11 July 2002
- Libe2_02 Liberty Alliance Project: Liberty Protocols and Schemas Specification, Version 1.0, 11 July 2002
- Libe4_02 Liberty Alliance Project: Liberty Bindings and Profiles Specification, Version 1.0, 11 July 2002
- Micr_01 Microsoft Corporation: Various .NET Passport documentation (started 1999), in particular Technical Overview, Sept. 2001, and SDK 2.1 Documentation; <http://www.passport.com> and <http://msdn.microsoft.com/downloads>
- PfWa_02 Birgit Pfitzmann, Michael Waidner: Privacy in Browser-Based Attribute Exchange; ACM Workshop on Privacy in the Electronic Society, Washington, Nov. 2002, post-conference proceedings to be published by ACM
- PfWa1_02 Birgit Pfitzmann, Michael Waidner: BBAE – A General Protocol for Browser-based Attribute Exchange; IBM Research Report RZ 3455 (# 93800), Sept 2002, <http://www.zurich.ibm.com/security/publications/2002.html>
- SAML_02 OASIS Security Assertion Markup Language (SAML); Committee specification 01, May 2002 (started Jan. 2001), <http://www.oasis-open.org/committees/security/docs>
- Shib_02 Shibboleth-Architecture DRAFT v05; May 2002 (v1 in 2001) <http://middleware.internet2.edu/shibboleth/docs/draft-internet2-shibboleth-arch-v05.pdf>
- Slem_01 Marc Slemko: Microsoft Passport to Trouble; Rev. 1.18, Nov. 2001 <http://alive.znep.com/~marcs/passport/>