

# Privacy in Browser-Based Attribute Exchange

Birgit Pfitzmann  
IBM Zurich Research Lab  
Säumerstrasse 4  
CH-8803 Rüschlikon, Switzerland  
bpf@zurich.ibm.com

Michael Waidner  
IBM Zurich Research Lab  
Säumerstrasse 4  
CH-8803 Rüschlikon, Switzerland  
wmi@zurich.ibm.com

## ABSTRACT

Browser-based attribute-exchange protocols enable users of normal web browsers to conveniently send attributes, such as authentication or demographic data, to web sites. Such protocols might become very common and almost mandatory in general consumer scenarios over the next few years. We derive the privacy requirements on such protocols from general privacy principles and study their consequences for the protocol design. We also survey to what extent proposals like Microsoft's Passport, IBM's e-Community Single Signon, SAML, Shibboleth, the Liberty Alliance specifications and a protocol BBAE of our own conform to these design consequences, and how one could go forward.

## Categories and Subject Descriptors

C2.0 [Computer-Communication Networks]: General – *Security and protection*; K4.1 [Computers and Society]: Public Policy Issues – *Privacy*; K4.4 [Computers and Society]: Electronic Commerce – *Security*.

## General Terms

Algorithms, Design, Security, Standardization, Legal Aspects.

## Keywords

Identity management, single signon, attribute-exchange, web browser, wallet, privacy, security, traffic data, roles, Passport, e-Community Single Signon, SAML, Shibboleth, Liberty, BBAE.

## 1. INTRODUCTION

Electronic commerce with end consumers has not fulfilled the high expectations of a few years ago. There are many possible explanations. By current consumer surveys, lack of trust, in particular with respect to privacy (here including protection from spam mail and unwanted phone calls) is the main reason for reticence. A second reason is that many services are not easy to use. A conclusion often drawn from this second observation is that

there should be single signon or one-click features across the entire world-wide web, i.e., transactions should essentially work without remembering anything and typing anything in, except typically for one master password and some OKs. However, unless designed very carefully, such features make the situation for trust and privacy even worse and are thus counterproductive. Nevertheless, analysts forecast that such protocols are likely to become very common over the next years.

Classically, single signon refers to authentication. However, in an e-commerce scenario, obtaining payment and shipping data for a transaction is even more important than authentication, and the sites are typically interested in additional demographic or usage data. Hence we talk of attribute exchange as a generalization of single signon.

A market observation currently taken for granted by the major players is that a large segment of users will not install additional software or even hardware for electronic commerce, neither for ease of use nor for security or privacy. Thus attribute-exchange protocols must accommodate this user segment, i.e., people using nothing but a commercial browser. We call such protocols browser-based, and the feature zero-footprint. A similar precondition is mobility in the sense that a user should be able to use the protocols from varying browsers, such as several personal devices or even Internet kiosks. While we stress that single signon from unknown browsers is dangerous for security, nobody is forced to do that (at least in developed countries), and thus we simply accept this as a market requirement. These functional requirements distinguish newer proposals like Microsoft's Passport, IBM's e-Community Single Signon, SAML, Shibboleth, and the Liberty Alliance specifications from classical form fillers, wallets, and PKIs.

Privacy surveys, e.g., [IBM99, Har02], show consistently that 80 to 90% of all people are concerned about privacy, and that 25% are willing to pay a considerable price in money or inconvenience for it. It seems to be well-accepted today that privacy concerns are a major inhibitor for e-commerce, as visible by privacy and trust initiatives by major software providers like IBM and Microsoft and the fact that privacy statements become common on e-commerce sites. It is also known that about half of all people at least occasionally give wrong data to web sites, and Gartner found that less than 10 percent of online consumers would be willing to exchange personal information in order to use personalized Web services. For browser-based attribute exchange, the experience with Microsoft Passport and the Hailstorm services (later called .NET my Services) based on it confirm this: Although Microsoft's privacy promises, at least after the first press reactions, are actually quite good, Gartner found that only 2 percent of consumers

chosed Passport for the convenience of single signon, compared with 16 percent six months earlier [Wil02]. Microsoft reacted on customer reluctance by shifting from offering Hailstorm services to consumers to offering software to other potential service providers. However, a mere change of provider will not alleviate all consumers' concerns: In a Gartner survey where people were asked how much they trusted different types of organizations with (rather uncritical) data like credit card number and address on a scale from 1 (low trust) to 7, no organization type got over 5 on average.

One consequence from these statistics in our analysis is that the same protocols for which we required a zero-footprint version should also work in a non-zero-footprint version, i.e., that the attributes to be exchanged can also reside on the user's own machine. To avoid misunderstandings: Clearly, a consumer *either* has a local wallet *or* a zero-footprint version. The goal is that the same protocols (in particular, with the same server-side software) work in both cases and offer the optimal combinations of ease-of-use, efficiency, and privacy within the limitations of each case. We will see that this is possible. Only such an approach with certain options (we will see further options besides local wallets versus zero footprint) allows e-commerce to satisfy both the 10-20% of consumers that certainly choose convenience over privacy and the 25% that certainly choose privacy over convenience. Moreover, it allows the remaining majority to develop their preferences without undue constraints. For instance, people in this half, in spite of their expressed privacy concerns, are known to sometimes release data quite globally and recklessly for convenience in a specific situation. They are likely to avoid that if they can easily restrict the release to that specific situation and the data needed there.

**Overview of this paper:** Chapter 2 presents the functional and security requirements that underlie current proposals of browser-based attribute-exchange protocols. In Chapter 3, we present the privacy requirements, which are our main concern. Chapter 4 gives an overview of protocols and products. In Chapter 5, we concentrate on browser-based protocols and derive technical consequences from the privacy requirements. In other words, we present design decisions that are mandatory to fulfil the privacy requirements. We also discuss to what extent the existing proposals fulfill them. We will see that while none of the current standards and deployed solutions fulfils them all (and actually none of them even claims to address the full functionality yet), current standards are a good basis for deriving very satisfactory protocols.

## 2. BROWSER-BASED ATTRIBUTE EXCHANGE

A browser-based attribute exchange protocol is a three-party protocol for information exchange, often including authentication, where the user only has a browser (in contrast to an arbitrary protocol engine as in "normal" protocol design).

### 2.1. Participants and Terminology

The scenario and some terminology are shown in Figure 1. A *user* is browsing at a *destination site*, e.g., a web shop. At some point, the destination site would like *attributes* about the user, such as a shipping address, credit-card number, or demographic data. The

goal of the protocol is that the user can send these attributes without having to remember them and to type them in.

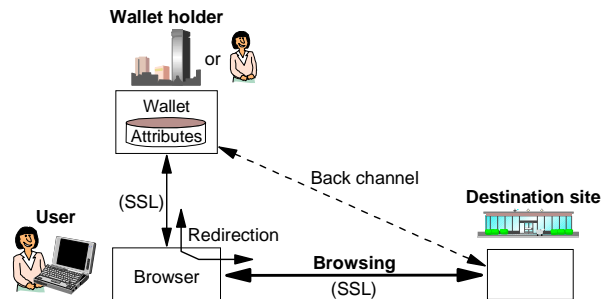


Figure 1. Scenario of browser-based attribute exchange

In the zero-footprint and mobility case, these attributes must obviously reside at some other party, called *wallet holder*. By a *wallet* of a user we mean the collection of attributes of this user at one wallet holder, together with the methods to use them. The prior phase where the wallet holder got the attributes is called *registration*; it may be based on a pre-existing relationship between the user and the wallet holder (e.g., a bank or an employer). The main mechanisms available for attribute transport in the zero-footprint case are browser redirection and SSL/TLS channels (used via HTTPS) for transport security. Additionally, a *back channel*, i.e., a direct channel between the wallet and the destination site may be established.

The easiest way to include *local wallets* is to run the same protocols with a wallet implemented on a machine of the user, typically the same machine where the browser resides. This is also shown in Figure 1. In contrast, *remote wallets* are wallets at third-party wallet holders.

### 2.2. Functionality and Infrastructure

We now summarize the functionality required of browser-based attribute-exchange protocols and some consequences on the design, including some that we already assumed in Figure 1.

- **Browser-based:** The protocols should be usable from all common web browsers on all common machines. (If this includes WAP, the usual WAP adaptations are needed.)
- **Case-by-case attribute exchange:** The protocols enable attribute exchange linked to a current browser session at a destination site, without direct authentication of the user at the destination site.

Consequently, in Figure 1 browser redirection is necessary even if a back channel is also used, because initially the destination site does not know about whom it wants attributes.

- **Cross trust domain:** The destination site and wallet holder may be in different trust domains.

This one cannot implement a remote wallet as a reverse proxy of the destination site for all users, and the back channel cannot be replaced by a database lookup.

- **Compatible with zero-footprint:** The protocols must be usable if a user has no software specifically for attribute exchange besides the browser, and has not enabled any active

content such as JavaScript, ActiveX and Java on the browser, e.g., for security reasons. They should even work for users who switch cookies off. (They may of course exploit active content or cookies where available to further increase ease of use.)

- **Compatible with mobility:** The protocols must accommodate users who move from device to device, e.g., by using Internet kiosks.

Thus the protocols cannot rely on browser personalization features (while they may use them for efficiency improvements where available).

- **Flexibility:** The protocols should be flexible. In particular, they should allow, all in one protocol superset:
  - Multiple wallet holders per user, e.g., a bank for financial information and a doctor for medical information. The wallets may or may not be linked. This will be a privacy requirement, but it is also convenient, and it is necessary if certain attributes need fresh certification by the relevant authorities.
  - User-chosen and certified attributes (e.g., book preferences versus a certified age).
  - Multiple roles per user. This will be a privacy requirement, but is also a functionality requirement, e.g., to distinguish payment and shipping information as a private user and as an employee.
  - Different user preferences regarding security and privacy and ease of use.
  - Both general-consumer and efficient closed-group usage.

We are mainly interested in short-term realizable protocols, without major infrastructure or hardware prerequisites. This mostly follows automatically from the requirements “browser-based” and “zero-footprint”. Additionally, new standardization should be minimized.

### 2.3. Security Requirements and Limitations

Although we concentrate on privacy, it is useful to be aware of the security requirements. Moreover, there are clear security limitations for browser-based attribute-exchange protocols, which should never be forgotten. The requirements are:

- **Authenticity:** If authentication to the destination site is intended – under a real name or a previously introduced role name or pseudonym – no adversary should be able to impersonate the user, not even with a man-in-the-middle attack.
- **Correctness and freshness of certified attributes:** Where attributes are confirmed by the wallet holder or another party, these confirmations should be unforgeable. Where desired, freshness should be guaranteed.

Accountability, in particular non-repudiation of the user, is currently not a goal, and it is not possible in the version with zero-footprint and mobility. Availability of the wallets is very important, but an issue of implementation more than of the protocols.

The main security limitations are the following:

- **Password security:** All authenticity relies on the wallet authenticating the user before sending names or attributes. This is typically done with a password, called single signon password, and for users choosing the mobility option nothing else is possible. For remote wallets, this makes the password available for online guessing. By trying this for enough users, an attacker is almost sure to be able to impersonate some users to their wallet holders, and thus to all destination sites using this protocol.
- **Browser and OS security:** A browser-based protocol cannot be more secure than the browser, at least in the zero-footprint version, nor than the underlying operating system. In particular, the single signon password passes through the browser and operating system. This is particularly dangerous for the mobility option, where the user uses unknown machines.

Other dangers are that a user is tricked into divulging the password, registration errors, and the usage of cookies, in the order of increasing amenability to design. Many details can be found in [KR00, Sle01] (see also [FSS+01]), but that only treats direct authentication to a destination site).

## 3. PRIVACY REQUIREMENTS

Concrete privacy requirements can be derived systematically from generally accepted principles like the following:

**Privacy principle:** Privacy is “the right of individuals to determine for themselves when, how, and to what extent information about them is communicated to others” [Wes67].

Nowadays one also generally requires that usage of the information is restricted to the purposes determined by the individual. On the other hand, one accepts the restriction that “determining” means at the minimum a visible policy of the “others” with opt-out choice for the individual, and that law may allow exceptions.

We now derive the privacy requirements on browser-based attribute-exchange protocols in a general consumer scenario.<sup>1</sup> The first consequence for attribute-exchange protocols is clear and universally accepted for this scenario, while the others are more easily forgotten.

### 3.1. Attribute Privacy [ATR]

A wallet  $W$  should only send attributes that it holds about a user  $U$  to a destination site  $D$  if  $U$  consented to that.

Moreover, in the general consumer scenario, where the wallet’s main role is as a trusted user agent, “determining” (as in the privacy principle) should be possible with a reasonable set of choices, so that users can determine that different destination sites get different subsets of the data, and some get no data at all.

---

<sup>1</sup> In closed environments with a-priori privacy restrictions, e.g., if the wallet holder is an employer and the protocol is only used for the employees’ interaction with partner companies, certain requirements can be relaxed. Good protocols automatically adapt to such environments, just as they adapt to users without privacy concerns.

### 3.2. Multiple Roles [ROL]

The privacy principle also applies to identifiers of the user  $U$ . For instance, even an identifier  $UID$  agreed upon between a wallet holder and  $U$ , without prior meaning, should not be sent to all destination sites automatically. In other words, a user should be allowed multiple unlinkable roles, and not only in the user-unfriendly way of multiple wallets.

Which destination sites accept users without identification under a real name, and for which actions, is another question. Anonymous or pseudonymous attribute exchange should be useful at least for web pages where one can currently browse without any identification, but where the site owners would like to ask for demographic or preference data.

### 3.3. Privacy of Traffic Data [TRA]

The privacy principle also applies to indirect information, in particular traffic data like the user's location and visited-URL trail.

- Wallet holders should not be allowed to mine or forward traffic data without consent.
- Wallet holders should not even *get* traffic data without consent or technical need. In particular, one should not implement a remote wallet as a user proxy, because a proxy sees the user's entire communication (except if it is part of an anonymizer chain). A remote wallet will learn the identities of the destination sites to which it sends attributes about a user  $U$ ; this seems unavoidable for the redirect back and for security against man-in-the-middle attacks. However, it should not learn the exact URLs of the pages a user visits.

### 3.4. Privacy from Wallet Holders [WAL]

The privacy principle also applies to the wallet holders as "others". Thus, no user  $U$  should be forced to trust a wallet holder  $W$  with attributes that are not already known there for some purpose. (Traffic data were already discussed in Section 3.3.) Therefore:

- **Wallet choice:** Zero-footprint users should be allowed a choice among different wallet holders. [WAL\_C]
- **Multiple wallets:** Each user should be allowed several wallet holders for different attributes, e.g., their bank for financial information and their employer for work-related information, and this should work seamlessly. (This was also a functional requirement.) [WAL\_M]
- **Local wallets:** Users should be allowed local wallets for attributes with whom they trust nobody, or at least nobody who offers wallet-holder services. A local wallet should be able to act independently (without allowing another party to impersonate it) and without leaking unnecessary information. [WAL\_L]

Although the local-wallet requirement is a natural consequence of the general privacy principle and leads to no problems in the protocols, people do not seem easy to convince people of it. We therefore give additional corroboration in Appendix A.

## 4. OVERVIEW OF PROTOCOLS AND PRODUCTS

To introduce the techniques used in browser-based attribute-exchange protocols, we sketch our own protocol BBAE, which fulfils all the privacy requirements. (More details can be found in [PW02]). We can then describe other protocols essentially as special cases.

By the functional requirements from Section 2.2, many older related proposals are not browser-based attribute exchange. For interested readers, we discuss them in Appendix B, both as prior art and because the same privacy requirements apply to them.

### 4.1. Example Protocol BBAE

An overview of a browser-based attribute-exchange protocols is given in Figure 2, compare also Figure 1.

- Before the start of the protocol, a user is browsing at a destination site (Step 0).
- In Steps 3-4, the destination site redirects the browser to the wallet; before that, it may have to find out the wallet's address (Steps 1-2).
- In Step 5, the wallet authenticates the user, unless they already have an authenticated session.
- Now a back channel may be used for actual requests and responses about attributes (Steps 6-7, 9-10). If the wallet does not know all attributes or the user has not set a policy that allows or forbids this release, the wallet may ask the user (Step 8); this is called real-time release.
- In Steps 11-12 the wallet redirects the browser back to the destination site.

Instead of using a back channel, one can integrate the attribute request into Steps 3-4 and the response into Steps 11-12. However, information transmitted in a redirect is typically put into the query string, which has length bounds that are easily exceeded if, e.g., the response is signed. (An alternative is a POST, but that requires either an additional user click or active content.)

Example screens for the user interaction in Steps 1-2 and 8 are shown in Appendix C.

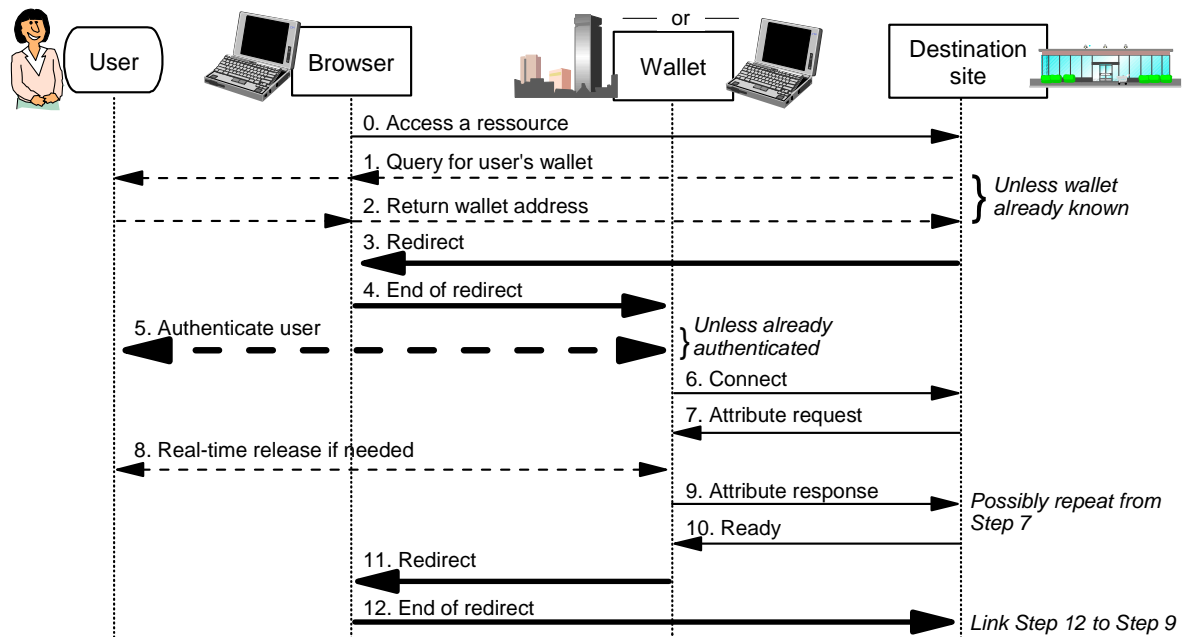
### 4.2. Proposals in Literature and Practice

In this section we sketch the known proposals and products for browser-based attribute exchange. More details are discussed in Section 5 in specific relation to privacy.<sup>2</sup> The last three protocols only address single signon; we include them because all four privacy requirements already apply for single signon alone.<sup>3</sup>

---

<sup>2</sup> Protocols with the same sequence of steps can differ in the contents of the query strings in the redirects, and the requests and responses. Some specifications also comprise additional phases like registration and logoff.

<sup>3</sup> The step between single signon and attribute exchange of just a name and address is also quite small. In particular, given a name one could set up independent back channels, so that some privacy policy should be in place.



**Figure 2. Browser-based attribute exchange. Bold-arrow steps occur in all protocols. Dashed-arrow steps are only needed in certain cases even in this protocol.**

**Microsoft Passport** is the best-known and to our knowledge first browser-based attribute exchange protocol (not fully published, but see [Mic01]). It does not use a back channel and assumes that the wallet is at a fixed Microsoft address. Thus it only contains Steps 3-5, 8, and 11-12. Authenticated channels with users (Step 5) are retained over several protocol instances with cookies. This was the main source of attacks (but is a natural tradeoff between security and ease of use). A limited set of attributes is handled with a fixed privacy policy: There are two data categories, demographics and purchase. Demographics consist of a fixed identifier and some attributes the user can enter or not, and are released to all destination sites that ask via this protocol. Purchase data need a real-time release per destination site, i.e., Step 8. A future version that allows different wallet holders – and thus probably adds Step 1-2 – and more flexible policies has been pre-announced in autumn 2001, but no details have been given yet, not even in [Mic02].

**SAML SSO.** SAML (Security Assertions Markup Language) [SAM02] is the main open standards proposal for browser-based attribute exchange. It is currently in a voting phase at OASIS, and the following Liberty Alliance specifications are also based on it. It primarily defines a message format for authentication and attribute assertions<sup>4</sup>, and for requests and responses about them. Assertions resemble certificates, but are XML-based and somewhat more flexible. They contain attributes and/or authentication

<sup>4</sup> It also contains authorization assertions, which can be used in the same protocols but do not specifically interest us here. Nor do we consider other specific digital-rights language proposals like XrML; anyway, they do not define browser-based exchange protocols.

information, issuer information, and possibly an XML “signature” [XML02] (which can be made with different techniques, even symmetric cryptography).

SAML is open for profiles, i.e., entire attribute-exchange protocols using SAML messages. The profiles standardized so far, called SAML SSO, are core parts for single signon, with the possibility to include additional attributes. They start when the browser is already pointed to the wallet, i.e., omit Steps 1-4, and explicitly assume Step 5 has taken place. Thus the profiles also apply if a portal acts as remote wallet, but for our general consumer scenario one needs a sub-profile for these steps. Then come Steps 11-12, 7, 9 in this order, i.e., first the browser is redirected back to the destination site, and then request and response are sent on a back channel set up by the destination site. A POST version has Steps 11-12 only.

**Shibboleth** [Shi02], an Internet2 project for a university community, can be seen as a powerful additional SAML profile if one extracts its protocol aspects. Different wallet holders are allowed and are typically the universities for their members. The main version consists of Steps 1-5, 11-12, 7, 9. Thus it starts with a query for the user’s university, redirection and user authentication as in Figure 2. Then, like SAML SSO, it first redirects back (except that this is here done with a POST) and then sends a request and response on a back channel. An addition for real-time release is a bit complicated, because the browser must again be redirected to the wallet, i.e., Steps 3-4 again, then Step 8 is possible, and then Steps 11-12, 7, 9 occur again.

[Go01] was an attempt, but seemingly unsuccessful, to get browser-based single signon standardized in IETF. The main protocol consists of Steps 1-5 and 11-12, i.e., the standard version without back channel. Back channels are also mentioned.

**IBM e-Community SSO** (single signon; not fully published, but see [IBM02], Ch. 10.2) has not been meant for a general consumer scenario, but for closer federations, so that some privacy requirements would be relaxed. The basic protocol is similar to Passport, i.e., Steps 3-5 and 11-12. Instead of one world-wide wallet holder, each community chooses its own wallet holder, called “master authentication server”, which is a parameter at the other community members, so that Steps 1-2 are not needed. Every destination site in a community can also act as a wallet (for authentication) when a user clicks on a link to another site in the community, by including authentication as in Step 12.

**Liberty Alliance.** The long-awaited first Liberty Alliance specifications [Lib02] are, like IBM e-Community SSO, explicitly restricted to single signon and close federations. (Extensions are announced for 2003.) They first extend SAML assertions, and then define profiles similar to, but not exactly extending the SAML SSO profiles. (In addition, [Lib02] contains a WAP profile and a non-browser profile, and a specification of authentication classes.) The basic profiles consist of Steps 3-5, 11-12, and then possibly 7, 9, i.e., the typical redirects back and forth with user authentication in between, and possibly a final use of a back channel (but only for the response). While the basic profiles leave open how the wallet is found (Steps 1-2), the only specified proposal is to transfer the information in cookies set on the browser for a joint domain of the federation, which is somewhat at odds with the functional requirement of cross-trust-domain operation.

## 5. DESIGN DECISIONS FOR PRIVACY

We now derive individual design decisions from the requirements in Section 3, and discuss to what extent these features are present in the proposals sketched in Section 4.2 and how they can be added. (Recall that no company or standards groups claims to have their final, general-consumer and privacy-friendly specification yet) We omit BBAE in this discussion because we have of course designed it to fulfil our requirements, and [Gol01] because it seems to have no chance as a standard and no unique features.

The requirements that imply the design decisions are given in parentheses in the headings.

Note that our main focus is on *protocols*, i.e., our discussion of policies and user interfaces is restricted to those aspects that have consequences on the protocol standards.

### 5.1. Flexible Privacy Policies (from [ATR])

A user should have flexible choices as to which destination sites may receive which attributes. Proposals for such policy languages and administration systems have, e.g., been made in [APP01, BLK+01]. Wallet holders that mainly administer information they hold anyway, like banks and employers, should use an enterprise privacy management system as in [KSW02].

Policies as such are an issue of set-up and interaction between wallet and user and thus directly visible only in systems proposals, not in protocol-standard proposals. Thus, for SAML one cannot speak of the presence or absence of policies. As mentioned, Microsoft Passport has a fixed simple policy with two data categories and no distinction among destination sites. (For one data category, it makes up for this with real-time release, see Section 5.2.) Shibboleth allows flexible policies, with a policy model consisting of attributes and data users. The latter can be names of

institutions and URL trees. The remaining proposals are only for single signon, i.e., “only” transfer identifiers, and are thus considered under consequences from [ROL] in Section 5.5; similar for the next three design decisions.

### 5.2. Integrate Real-time Release (from [ATR])

Policies are complicated. Hence at least in the short run, people cannot set a privacy policy in advance that classifies all cases adequately. Therefore real-time release of attributes will play an important role. In policies this shows up as an additional decision “ask me” besides “allowed” and “forbidden”, and in the protocol as a query to the user; see Step 8 of Figure 2. Real-time release is much easier in browser-based protocols than in other uses of privacy policies because the user is already involved. There is no problem that the user might be bothered too much:

- Even a protocol without any policy, only with real-time release, is more convenient than the present situation (only clicks for “OK” or “no”, instead of typing data in).
- People who do know their policies in advance can set them and will never be bothered again (in particular, people who want to allow all releases).
- User studies in many fields show that people prefer to start with examples, not with abstract concepts. Thus they are more likely to quickly arrive at policies they like if policy development is initiated by real-time releases. Such efforts are under way, e.g., in the IBM Privacy Services project (not yet described in [BLK+01]).

Recall from Section 4.2 that Passport (for its purchase category) and Shibboleth (as an exception for arbitrary data) include real-time releases, while SAML does not consider this aspect.

### 5.3. Authenticate Destination Site (from [ATR])

A privacy policy usually allows the release of certain attributes to some destination sites and not to others. Hence the destination sites must be authenticated in the exchange protocol. A minimum is authentication by name; in principle arbitrary attributes are conceivable.<sup>5</sup>

- In protocols with back channel, the wallet  $W$  can authenticate the destination site  $D$  in standard ways, e.g., with an SSL server certificate if  $D$  is the server for the back channel as in Figure 2. If a real-time release is needed,  $W$  can then also securely tell the user  $U$  to whom the release will go. (This is indicated by  $\langle name \rangle$  in the user interface example in Figure 4 in Appendix C).
- Without back channel, one either has to rely on the user verifying with whom she is communicating and compare this

---

<sup>5</sup> For elegant privacy policies, an infrastructure for attribute-based authentication would be useful, e.g., to recognize a destination site as a bank, a travel agency, or a site with a certain privacy seal, but we do not see this happening for short-term deployment. Furthermore, most users will want to distinguish *their* bank, travel agency etc.

with the wallet's belief, or one needs end-to-end protection between wallet and destination site.

Shibboleth has a back channel and therefore uses the former measure. As we saw in Section 5.1, most other proposals do not have policies that distinguish destination sites yet and thus need not worry. SAML can certainly be instantiated in this way. For Passport's real-time release, the employed end-to-end encryption seems to help (but recall that the protocol is unpublished).

#### 5.4. Policy Information in Request/Response (from [ATR])

In privacy policies, a release is typically for specific purposes and with obligations such as future deletion. A wallet and a destination site must negotiate this. The easiest way is to enable the inclusion of privacy promises in attribute requests, and of privacy restrictions in the responses. Lengthy policies may make the use of a back channel mandatory. A repetition of Steps 7-9 of Figure 2 can then be used for more complicated negotiations.

The systems proposals, Passport and Shibboleth, do not have this yet. SAML does not mention it, but allows attributes with arbitrary types in assertions. Hence one could standardize a type that contains a privacy policy for a sub-profile.

#### 5.5. Policies for Identifiers (from [ROL])

By the multiple-roles requirement, no identifier should be sent without user consent. The easiest solution is that a wallet handles identifiers like all other attributes.<sup>6</sup> In particular, identifiers would occur only in Step 9 of Figure 2, which underlies the policies, and nothing like a distinguished name should be mandatory in the assertion format.

Shibboleth achieves the same effect by first only sending a handle (in Steps 11-12, which occur before Steps 7 and 9 there), which is recommended to be a nonce. Passport, in contrast, sends a global identifier on each request. Destination sites should only make a request if the user clicks a button, and otherwise, user authentication (Step 5) takes place if the user has no current session with the wallet. Hence the user may notice unauthorized requests, which should deter most destination sites. Nevertheless, one can hope that the next version really allows multiple roles (beyond having multiple wallets, which Passport explicitly allows). SAML leaves the issue open.

For the single-signon-only proposals, we must consider the identifiers alone. IBM e-Community SSO does not consider privacy, because in its typical closed-federation applications, linking the identities in a federation is the goal and the rights to do that should be given by other contracts. For wider usage, one could

---

<sup>6</sup> If a user wants to exploit this beyond being anonymous in some cases and identified in others, the wallet must offer a user interface for managing roles, i.e., different identifiers (pseudonyms) and partially different attributes. Besides nonces and user-memorable pseudonyms, there is the proposal of [GGK+99] to derive them by a fixed pseudorandom function from a partner identity. This saves storage, but only if no other attributes are different in different roles, and it is a problem if one wants to interact with one destination site in two roles.

add policies as to the other solutions. Liberty's goal is also to link identities. It unfortunately does not clarify whether linking always implies that additional attributes can be exchanged outside the protocol, or never, or according to contracts in the federation and how the user could study those. Liberty solves the functional problem of wallets and destination sites to map preexisting user identities to each other by letting the user do this initially; this phase requires user consent and can be revoked. The consent for a specific destination site is only given at that destination site, and the destination site can prohibit user interaction at the wallet, so that the user would not notice illegitimate attempts. For a general consumer scenario, this would have to be changed. The identifiers that a wallet releases do not enable linking of identities at different destination sites. If the cookie-version of finding a wallet is used (recall Section 4.2) and the user retains these cookies, follow-on authentication typically happens without user interaction. The joint domain set up for these cookies is also a privacy problem because it can be used for unintended cross-domain cookies.

#### 5.6. Minimum Information by URLs (from [TRA])

To prevent remote wallet holders from getting unnecessary trails of a user's browsing behavior, a destination site should not divulge the URL of the user's original request (Step 0).

With most protocols, this may happen as follows: This *target URL* is included in the query string of the redirect in Steps 3-4, and later used to redirect the user back in Steps 11-12. Although the choice of these parameters is outside the actual specifications, Passport, SAML, Shibboleth and Liberty all suggest this implementation. Shibboleth also shares the target URL with a possibly central lookup service of wallet names. IBM e-Community SSO leaves it open.

Instead, the destination site should use a fixed URL and a random session ID to link the different sub-sessions together. While this can be done outside the standard by internal redirection, we believe that at least a privacy-friendly recommendation should be given in all standards. Further, deviations should have to be mentioned in privacy policies of the site, i.e., sending target URLs to wallets should not count as part of the current purpose.

#### 5.7. Query for Wallet Holder (from [WAL\_C, WAL\_M])

Free choice of a remote wallet holder [WAL\_C] implies that a destination site does not a-priori know the wallet holders of all its users, in contrast to Microsoft Passport and close-federation solutions like IBM e-Community SSO. Thus a query for the user's wallet location (Steps 1-2 of Figure 2) is necessary. The user in person must be asked at least in the mobility case, or if all active content and cookies are switched off, because the browser then also knows nothing about the user. Allowing multiple unlinked wallets per user [WAL\_M] makes this case even more common; at least destination sites that try to derive the wallet location in other ways must provide error handling in case they arrived at the wrong wallet of a user.

We already mentioned in Section 4.2 which proposals have these steps.

## 5.8. Hiding the Wallet Address (from [WAL\_L, ROL])

A local wallet, and thus its user, can also be identified by its address. The protocol should therefore be designed without needing this address.<sup>7</sup> (This only works if the wallet is on the same machine as the browser; using a home wallet from a kiosk browser needs the address for the initial redirect in Steps 3-4.)

- **Redirects to localhost.** The redirect to the wallet (Steps 3-4 of Figure 2) must be to a path at localhost. This only means that a standard should not make unnecessary requirements on the address format; redirects automatically also work for localhost.
- **Back channels established by wallet.** In protocols with back channels, the wallet should contact the destination site, not vice versa, i.e., Step 6 of Figure 2 is indeed needed.

None of the current proposals with back channels fulfils this; see the description of their steps in Section 4.2. Questions about redirects to localhost can only be asked for protocols that already contain a choice of wallets, i.e., Shibboleth only. There the lookup service for wallet addresses would have to be extended by this address.

## 5.9. Allow Public-key Cryptography (from [WAL\_L, TRA])

The local-wallet requirement includes that a local wallet can act independently. Hence the protocol must support public-key cryptography: Local wallets cannot exchange symmetric keys with all potential destination sites in advance and using a key- or ticket-distribution center (e.g., as in Kerberos) allows the center to impersonate the wallet and gives it an unnecessary usage trail.<sup>8</sup> We recommend that remote wallet holders also use public-key systems or directly exchanged symmetric keys, so that they can also act independently.

This is currently not possible with Passport, and the planned change to standard Kerberos tickets as assertions would make no difference.<sup>9</sup> SAML and Shibboleth allow it by only referring to

---

<sup>7</sup> Hiding the wallet address in the attribute-exchange protocol is useful even though lower layers may identify a user: First, lower-layer information may be concentrated in fewer modules in the destination site and thus less likely to be divulged accidentally. Secondly, if the lower layers provide some anonymization, from dynamic addresses assigned by the IP provider to complicated mixes [Cha81], anonymous addressing is not always implemented, while anonymous responses over a given connection usually are.

<sup>8</sup> For sending attributes without any identifier, e.g., anonymous demographics, no key at all is needed, and for roles used longer-term but with only one partner, the wallet could generate a fresh symmetric key. However, some situations will need a confirmed identity and thus either a direct key exchange or a certificate or secret-key ticket.

<sup>9</sup> Using the real Kerberos *protocol* would be yet another issue. It would not be a browser-based protocol because current standard browsers do not support it, and for the privacy and impersona-

XML signatures [XML02], which can be real signatures or message-authentication codes. Liberty uses the same schema but prescribes public-key cryptography in the text requirements. In e-Community SSO it is not mentioned, but given the closed federations and no back channels, secret keys only are likely.

## 5.10. Unknown Issuers Possible (from [WAL\_L, ROL])

Attributes sent by local wallets must be accepted by destination sites, both with and without authentication.

In situations without any identifier, attribute responses without “issuer” fields and signatures are simplest. Repeated usage of a role without outside meaning is best handled by a new issuer name and role key without a certificate, and acting in a “real” identity by a certified wallet key. (Remote wallet holders can always use the same key if their user group is sufficiently large for anonymity.) In addition, individual attributes can be certified, as long as the attribute certifier and the destination site may see the user in the same role. This can only be improved upon by unlinkable credentials, see Appendix B.

All current proposals need extensions to fulfill this. For current Passport and e-Community SSO, with fixed remote wallets and secret-key cryptography, we need not even discuss this point. SAML *assertions* are suitable because they make no requirements on issuer names, and XML signatures can contain arbitrary key information or certificate chains. However, the SAML SSO profiles requires destination sites to be able to look up URLs of wallet holders from IDs. These should instead be securely sent within the protocol. The same holds for Liberty. Shibboleth, intended for universities as wallet holders, requires inclusion of a domain name in the name-identifier element of the SAML assertion, which would at least need to be interpreted widely.

## 6. SUMMARY

Browser-based attribute-exchange protocols can offer convenience to web users and seem widely desired by enterprises. We have investigated the privacy requirements when such protocols are used in a general consumer scenario, as they follow from general, accepted privacy principles. In particular, no automatic identification under any fixed identifier should occur, no unnecessary traffic data should arise, and local and remote wallets should be able to coexist in one protocol. We then showed technical decisions implied by these requirements, most notably that public-key cryptography must be included and how relative addresses of local wallets can be sufficient.

The sketch of our protocol BBAE from [PW02] in Figure 2 shows that an all-purpose protocol that includes these decisions can be very efficient, and implies no inconvenience for the small part of the population without privacy concerns, because all privacy features are optional on the user side.

---

tability reasons, extending all browsers by Kerberos would be a step in the wrong direction.

## 7. OUTLOOK

The individual requirements and design decisions hopefully help in future developments and standardization; recall that nobody yet claims to have their final system, in particular neither Microsoft nor the Liberty Alliance.

The requirements can also serve as objectives in evaluations, and the design decisions as functional requirements.

For destination sites and remote wallets, a privacy-protecting browser-based attribute-exchange protocol is no harder to integrate than any other, beyond the necessary ability to locally enforce privacy promises at all. Local wallets, however, are not an attractive business opportunity, and a standard that allows them is no guarantee to get them. One may hope for the open-source community, or that they come with the desktop or browser in a trust-building initiative of a major company<sup>10</sup>, or that companies interested in trust by end customers, e.g., some banks, offer them at a reasonable price. All this, however, this needs an additional awareness campaign among consumers about the risks and benefits of the variants of browser-based attribute exchange.

## Acknowledgments

We have benefited from discussions with many colleagues, in particular Peter Buhler, Peter Capek, Chris Giblin, Thomas Groß, John Hind, Stephen Levy, Matthias Schunter, and Jay Unger.

## References

- APP01 A P3P Preference Exchange Language 1.0 (APPEL1.0); W3C Working Draft 26 February 2001, <http://www.w3.org/TR/P3P-preferences.html>
- BLK+01 Kathy Bohrer, Xuan Liu, Dogan Kesdogan, Edith Schonberg, Moninder Singh, Susan L. Spraragen: Personal Information Management and Distribution; 4th Intern. Conf. on Electronic Commerce Research (ICECR-4), Dallas, Nov. 2001
- Cha81 David Chaum: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms; Communications of the ACM 24/2 (1981) 84-88
- Cha85 David Chaum: Security without Identification: Transaction Systems to make Big Brother Obsolete; Communications of the ACM 28/10 (1985) 1030-1044
- CL00 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
- CV02 Jan Camenisch, Els Van Herreweghen: Design and Implementation of the Idemix Anonymous Credential System; to appear at ACM CCS 2002, Washington, Nov. 2002
- DH76 Whitfield Diffie, Martin E. Hellman: New Directions in Cryptography; IEEE Transactions on Information Theory 22/6 (1976) 644-654
- FSS+01 Kevin Fu, Emil Sit, Kendra Smith, Nick Feamster: Dos and Don'ts of Client Authentication on the Web; Proc. 10th USENIX Security Symposium, 2001
- Gat99 Gator: The Smart Online Companion; first release 1999, <http://www.gator.com/>
- GGK+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
- Gol01 Y. Y. Golang: Zero Install Single Sign On Solution for a HTTP Browser; Internet Draft, Nov. 2001, <http://www.ietf.cnri.reston.va.us/internet-drafts/draft-golang-ss-human-00.txt>
- Har02 Harris Interactive: First Major Post-9/11 Privacy Survey Finds Consumers Demanding Companies Do More To Protect Privacy; Rochester, Feb. 2002, <http://www.harrisinteractive.com/news/allnewsbydate.asp?NewsID=429>
- HTT99 Hypertext Transfer Protocol – HTTP/1.1; Internet RFC 2616, 1999
- IBM97 IBM Consumer Wallet; first release 1997, White Paper 1999, <http://www-3.ibm.com/software/webservers/commerce/payment/wallet.pdf>
- IBM99 IBM Multi-National Consumer Privacy Survey, conducted by Louis Harris & Associates, Inc.; IBM Global Services, October 1999
- IBM02 IBM: Enterprise Security Architecture using IBM Tivoli Security Solutions; April 2002, <http://www.redbooks.ibm.com/abstracts/sg246014.html>
- IM02 IBM Corporation, Microsoft: Security in a Web Services World: A Proposed Architecture and Roadmap, V 1.0; April 2002, <http://www-106.ibm.com/developerworks/library/ws-secmap/>
- KR00 David P. Kormann, Aviel D. Rubin: Risks of the Passport Single Signon Protocol; Computer Networks 33 (2001) 51-58
- KSW02 Günter Karjoth, Matthias Schunter, Michael Waidner: Platform for Enterprise Privacy Practices; to appear in these proceedings.
- Lib02 Liberty Alliance Project (founded 2001): Specifications Version 1.0, July 2002, <http://www.projectliberty.org/specs/liberty-specifications-v1.0.zip>
- Mic01 Microsoft Corporation: .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>

---

<sup>10</sup> This opens the question whether one trusts the operating system, and if one does, why one would not also trust remote wallets by the same company. Better evaluation possibilities are one answer.

- Mic02 Microsoft Corporation: Microsoft Federated Security and Identity Roadmap, June 2002, <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwebsrv/html/wsfederate.asp?frame=true>
- P3P02 The Platform for Privacy Preferences 1.0 (P3P1.0) Specification; W3C Recommendation, April 2002, <http://www.w3.org/TR/2002/REC-P3P-20020416/>
- Pas99 Passlogix: v-Go Single Signon; first release 1999, White Paper 2000, [http://www.passlogix.com/media/pdfs/usable\\_security.pdf](http://www.passlogix.com/media/pdfs/usable_security.pdf)
- PKI02 Public-Key Infrastructure (X.509) Working Group: An Internet Attribute Certificate Profile for Authorization; RFC 3281, 2002, <http://www.ietf.org/rfc/rfc3281.txt>
- PW02 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>
- Rob99 Roboform: Free Web Form Filler and Password Manager; first release 1999, <http://www.siber.com/roboform/>
- SAM02 OASIS Security Assertion Markup Language (SAML); Committee specification 01, May 2002 (started Jan. 2001), <http://www.oasis-open.org/committees/security/docs>
- Shi02 Shibboleth-Architecture DRAFT v05; May 2002 (v1 in 2001) <http://middleware.internet2.edu/shibboleth/docs/draft-internet2-shibboleth-arch-v05.pdf>
- Sle01 Marc Slemko: Microsoft Passport to Trouble; Rev. 1.18, Nov. 5, 2001 <http://alive.znep.com/~marcs/passport/>
- Wes67 Alan F. Westin: Privacy and Freedom; Atheneum, New York NY, 1967
- Wil02 Joe Wilcox: Customers wary of online IDs and Survey: Passport required—not appealing, CNET News.com, April 2002, <http://news.com.com/2100-1001-892808.html> and <http://news.com.com/2100-1001-884730.html>
- XML02 XML-Signature Syntax and Processing; W3C Recommendation, Feb. 2002, <http://www.w3.org/TR/xmlsig-core/>
- Zer99 Zeroknowledge: Freedom Personal Firewall; first release 1999, <http://www.freedom.net/products/firewall/index.html>

## Appendix A. More Arguments for Allowing Local Wallets

Although allowing local wallets to users who want them is a natural consequence of general privacy principles, and leads to no special problems in the protocols, it seems quite hard to convince people of it. We therefore give additional corroboration. Recall that inclusion of local wallets, as we are arguing for, means that

- the same protocol standard accommodates users who prefer the zero-footprint option and those who prefer a local wallet,<sup>11</sup> and
- no inconvenience results for local-wallet users beyond the need to install and set up the local wallet, so that they do not pay an unreasonable price for privacy.

First, local wallets are also important for authenticity: A wallet holder can impersonate a user. This is technically more like a power of attorney than an ID card or other typical analogs, and not all users will want to grant that in exchange for convenience and the wallet holder’s promise not to exercise it without a user OK. Being *forced* to give a power of attorney essentially corresponds to being put under guardianship.

Secondly, the surveys cited in the introduction show that currently only a small minority trusts any organization as a holder of wallets even with moderate functionality. Wallet holders might argue that people *should* trust them if they promise to keep the data in a secure environment. However, besides the contradiction in “having to trust”, insider fraud is currently not rare even in the financial sector. In the future, wallet holders might put the entire wallet functionality in tamper-resistant hardware without any access rights for the company. This would be very beneficial against insider fraud. However, we believe that no-one would seriously claim that hardware currently exists that can withstand a determined attack by a wallet holder as such (necessarily a rich and technology-informed organization), and we expect it never will.

Thirdly, one hears the argument that those who do not want a remote wallet could have no wallet at all. But, besides the resulting inconvenience, even one quasi-monopoly provider of an important service can make the usage of *some* wallet mandatory. While users may limit potential privacy violations by minimum usage of that wallet, they automatically become vulnerable to impersonation at all destination sites that accept this protocol for real-world names or addresses.

## Appendix B. Remotely Related Proposals

The functional requirements from Section 2.2 mean that many related proposals do not fall into the class of browser-based attribute exchange. We nevertheless briefly mention them, both as prior art and because the privacy requirements on them are similar.

**Form fillers:** Classical form fillers are user proxies that capture and try to fill forms that ask for data, e.g., [Rob99, Gat99]. The proxies may be local or remote. While remote proxies almost have the zero-footprint and mobility functionality, they have great privacy disadvantages (Section 3.3). Besides, with current attribute-exchange protocols one tries to agree on a standard for attribute

---

<sup>11</sup> Whether the two cases are handled in the same “protocol” is merely a terminology discussion. The important aspect is that both are mandatory in the same standard. For the simple case as needed in current e-commerce scenarios, a joint protocol with only minor variants will do, as this paper shows.

queries, so that the difficult task of analyzing arbitrary forms for known attributes becomes obsolete.

**Classical wallets** were designed either as remote user proxies or as browser plug-ins [IBM97, Pas99, Zer99]. Thus they do not have a zero-footprint version with reasonable privacy, see Section 3.3. (Browser plug-ins are a valid way to implement local wallets; however, local wallets as servers are less browser-dependent and simpler in the given class of protocols.)

**In-enterprise single signon** typically uses a reverse proxy for the enterprise, and applications in the enterprise can use a common registry to look up attributes (e.g., see [IBM02], Ch. 6). This does not fit the case-by-case and cross-trust-domain requirements of the general consumer scenario. However, these solutions can serve as the destination sites' implementation basis for browser-based attribute-exchange protocols.

**PKIs and attribute certificates:** For mere authentication, digital signatures [DH76] are technically the best single-signon choice, but here our main concern is attribute exchange. Attribute certificates as in [PKI02] can be used, but still an actual exchange protocol has to be designed, in particular for transmitting certificates with unmodified browsers. (Essentially, one of the browser-based protocols could be used. In other words, certain messages in these protocols are similar to attribute certificates; only the proposed message formats are different and based directly on XML.) Furthermore, most attributes in general e-commerce do not need certification, e.g., shipping address and user preferences.

**Unlinkable credentials:** From the long-term research side, unlinkable credentials are related, as invented in [Cha85] and currently developed furthest in [CL01, CV02]. They need a local wallet as browser proxy. Moreover, as privacy-enhanced versions of attribute certificates, they are overkill for typical e-commerce attributes: Most attributes are either identifying (like a shipping address) or do not need certification (like user preferences). Even payment information, which would theoretically need privacy and certification, is currently simply not realistic in that form. In this paper, we study these simpler cases. However, where certification and strong privacy *are* desired, only unlinkable credentials will help.

**WS Roadmap:** The Web services security roadmap by IBM and Microsoft [IM02] mentions identity federations, however only a web services scenario and no browser scenarios are considered.

## Appendix C. Example User Interaction

A possible form for asking a user for a wallet location, i.e., Steps 1-2 of Figure 2, is shown in Figure 3. The radio button "it is local" leads to a redirect to localhost (see Section 5.8).

We would like to know something about you.  
How can we find out?  
 No, I don't want to tell anything  
 I have a user name/password with you  
 I have a wallet and  
 ... it is local  
 ... my wallet holder is   
  
You can see [who we are](#) and our [privacy policy](#).

Figure 3. Example form for a wallet location query (Step 1)

An (already fairly complicated) example of a real-time release, i.e., Step 8 of Figure 2, is shown in Figure 4. All fields are editable so that the user can add or delete individual attributes.

Your partner, <name>, would like the following data about you:  
• Name  ★  
• Shipping address  ★  
• National ID number   
• Your income   
~~X~~ You probably don't want to send this  
★: They won't continue without this

Figure 4. Example form for a real-time release (Step 8)