

---

Title: **Information about the Liberty Alliance Project**

Source: **Nokia, Siemens, Vodafone**

Document for: **Information, discussion and decision**

## Introduction

This document is to inform 3GPP SA about the Liberty Alliance Project and its activities regarding an open, interoperable standard for federated network identity. It is realized that 3GPP is currently investigating and developing several issues, which are related to Liberty's activities, such as subscriber certificates, general user profile, and user privacy enhancements in location services, i.e. pseudonyms to hide the identity of the target mobile and the enhanced codeword to protect the target mobile user. The "enhanced codeword" is actually quite close to "Service Authorization and Authentication".

This document offers information about Liberty, which hopefully will be found quite relevant when developing corresponding issues in 3GPP. This document contains a general introduction regarding identities and the scope of the Liberty Alliance Project as described at <http://www.projectliberty.org/>. A summary document describing the Liberty Architecture version 1 and a slide set presenting the preliminary content of Liberty version 2 are also attached.

## Proposal

It is proposed that 3GPP SA investigates the activities and deliverables of the Liberty Alliance Project. If these are found to be relevant for 3GPP there should be a formal Liaison established between 3GPP and the Liberty Alliance Project to enable information sharing, feedback and reuse of existing standards. One approach could be to make references both-way between 3GPP Technical Specifications and Liberty Specifications in order to avoid lengthy and potentially conflicting duplications of specifications.

## Network identities

The basic element of a community is the notion of identity. Technology exists today to create, manage, and authenticate online identities and broker services based on information related to that identity. The Liberty Alliance Project is an alliance formed to deliver and support a federated network identity solution for the Internet that enables single sign-on for consumers as well as business users in an open, federated way.

Federated network identity represents the natural evolution of the next generation of the Internet. The first waves of the Internet, namely communications, global access, commerce, and community identity, gave us a pervasive user medium that has largely been relegated to one-to-one, customer-to-business relationships and experiences. The inflection point starting the Internet's next wave will be marked by an era of open, federated identity with promises of bold new business taxonomies and opportunities, coupled with economies of scale that, until recently, were simply unimaginable.

Federated network identity will enable the next generation of the Internet: federated commerce. In a federated view of the world, a person's online identity, their personal profile, personalized online configurations, buying habits and history, and shopping preferences are administered by users, yet securely shared with the organizations of their choosing. A federated network identity model will enable

every business or user to manage their own data, and ensure that the use of critical personal information is managed and distributed by the appropriate parties, rather than a central authority.

## Scope of the Liberty Alliance Project

The role of the Liberty Alliance Project in all of this is to support the development, deployment and evolution of an open, interoperable standard for federated network identity.

The vision of the Liberty Alliance is to enable a networked world in which individuals and businesses can more easily conduct transactions while protecting the privacy and security of vital identity information. To accomplish its vision, the Liberty Alliance will establish an open standard for federated network identity through open technical specifications that will:

- Support a broad range of identity-based products and services
- Enable commercial and non-commercial organizations to realize new revenue and cost saving opportunities that economically leverage their relationships with customers, business partners, and employees
- Provide consumers with choice of identity provider(s), the ability to link accounts through account federation, and the convenience of single sign-on, when using any network of connected services and devices
- Increase ease of use for e-commerce consumers
- Help to stimulate e-commerce

The key objectives of the Liberty Alliance are to:

- Develop specifications that enable service providers to protect consumer privacy
- Provide an open single sign-on specification that includes federated authentication from multiple providers operating independently
- Enable commercial and non-commercial organizations to control, maintain and enhance relationships
- Create a network identity infrastructure that supports all current and emerging network access devices

The Liberty Alliance is comprised of over 140 member companies representing a wide variety of industries and over a billion customers, with operations all over the globe. Each of the member companies either owns and operates large communities of interest or is the developer of core technology that can enable a federation of online communities. However, membership in the Alliance is still open and all organizations are invited and encouraged to join.

## Attachments

### 1. Liberty Architecture (version 1.0)

In the first attachment is a summary presentation of the Liberty version 1.0 Architecture.

### 2. Liberty version 2.0

Liberty version 2.0 architecture work is ongoing. In the second attachment is a preliminary content of the Liberty version 2.0.

# **Liberty Alliance Project**

*Building an Open Standard  
for Network Identity*

## Approach Drivers

- **Support rapid acceptance and deployment**
- **Phases build on each other**
- **Easy incremental adoption**



### **Version 1.0 (Released 07/15/02)**

- Federated network identity
- Opt-in account linking and simplified sign-on within an authentication domain created by business agreements
- Security built across all the features and specifications

### **Future Versions**

- Permissions-based attribute sharing
- Schema/protocols for core identity profile service
- Simplified sign-on across authentication domains created in version 1.0 by business agreements
- Delegation of authority to federate identities/accounts

- **Permissions-Based Attribute Sharing**
  - Enable businesses to share a principal's attributes according to their corporate policies, business agreements and local regulations, all while adhering to the principal's preferences and permissions
  
- **Interoperability Specs for Core Identity Profile Service**
  - Enables users to obtain secure, personalized services that are interoperable across different service providers
  
- **Federation of Authentication Domains**
  - Enables users to conveniently navigate and use SSO and share attributes with service providers who may be in different authentication domains.

Version 2.0 specifications expected Q1 2003

# LIBERTY ALLIANCE project

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30

## Liberty Architecture Overview

Version 1.0

11 July 2002

**Document Description:** liberty-architecture-overview-v1.0

30 **Notice**

31

32 Copyright © 2002 ActivCard; American Express Travel Related Services; America Online, Inc.; Bank of  
33 America; Bell Canada; Catavault; Cingular Wireless; Cisco Systems, Inc.; Citigroup; Cyberun Corporation;  
34 Deloitte & Touche LLP; EarthLink, Inc.; Electronic Data Systems, Inc.; Entrust, Inc.; Ericsson; Fidelity  
35 Investments; France Telecom; Gemplus; General Motors; Hewlett-Packard Company; i2 Technologies, Inc.;  
36 Intuit Inc.; MasterCard International; Nextel Communications; Nippon Telegraph and Telephone Company;  
37 Nokia Corporation; Novell, Inc.; NTT DoCoMo, Inc.; OneName Corporation; Openwave Systems Inc.;  
38 PricewaterhouseCoopers LLP; Register.com; RSA Security Inc; Sabre Holdings Corporation; SAP AG;  
39 SchlumbergerSema; Sony Corporation; Sun Microsystems, Inc.; United Airlines; VeriSign, Inc.; Visa  
40 International; Vodafone Group Plc; Wave Systems. All rights reserved.

41

42 This Specification has been prepared by Sponsors of the Liberty Alliance. Permission is hereby granted to use  
43 the Specification solely for the purpose of implementing the Specification. No rights are granted to prepare  
44 derivative works of this Specification. Entities seeking permission to reproduce portions of this document for  
45 other uses must contact the Liberty Alliance to determine whether an appropriate license for such use is  
46 available.

47

48 Implementation of this Specification may involve the use of one or more of the following United States  
49 Patents claimed by AOL Time Warner, Inc.: No.5,774,670, No.6,134,592, No.5,826,242, No. 5,825,890, and  
50 No.5,671,279. The Sponsors of the Specification take no position concerning the evidence, validity or scope  
51 of the claimed subject matter of the aforementioned patents. Implementation of certain elements of this  
52 Specification may also require licenses under third party intellectual property rights other than those identified  
53 above, including without limitation, patent rights. The Sponsors of the Specification are not and shall not be  
54 held responsible in any manner for identifying or failing to identify any or all such intellectual property rights  
55 that may be involved in the implementation of the Specification.

56

57 **This Specification is provided "AS IS", and no participant in the Liberty Alliance makes any warranty**  
58 **of any kind, express or implied, including any implied warranties of merchantability, non-infringement**  
59 **or third party intellectual property rights, and fitness for a particular purpose.**

60

61 Liberty Alliance Project  
62 Licensing Administrator  
63 c/o IEEE-ISTO  
64 445 Hoes Lane, P.O. Box 1331  
65 Piscataway, NJ 08855-1331, USA

66



66 **Editor**

67 Jeff Hodges, Sun Microsystems, Inc.

68 **Contributors**

69  
70 The following Liberty Alliance Project Sponsor companies contributed to the development of this  
71 specification:  
72

|  |  |
|--|--|
| ActivCard                                | MasterCard International               |
| American Express Travel Related Services | Nextel Communications                  |
| America Online, Inc.                     | Nippon Telegraph and Telephone Company |
| Bank of America                          | Nokia Corporation                      |
| Bell Canada                              | Novell, Inc.                           |
| Catavault                                | NTT DoCoMo, Inc.                       |
| Cingular Wireless                        | OneName Corporation                    |
| Cisco Systems, Inc.                      | Openwave Systems Inc.                  |
| Citigroup                                | PricewaterhouseCoopers LLP             |
| Cyberun Corporation                      | Register.com                           |
| Deloitte & Touche LLP                    | RSA Security Inc                       |
| EarthLink, Inc.                          | Sabre Holdings Corporation             |
| Electronic Data Systems, Inc.            | SAP AG                                 |
| Entrust, Inc.                            | SchlumbergerSema                       |
| Ericsson                                 | Sony Corporation                       |
| Fidelity Investments                     | Sun Microsystems, Inc.                 |
| France Telecom                           | United Airlines                        |
| Gemplus                                  | VeriSign, Inc.                         |
| General Motors                           | Visa International                     |
| Hewlett-Packard Company                  | Vodafone Group Plc                     |
| i2 Technologies, Inc.                    | Wave Systems                           |
| Intuit Inc.                              |  |

73

74

74 **Table of Contents**

|     |       |  |    |
|-----|-------|--|----|
| 75  | 1     | Introduction   | 5  |
| 76  | 1.1   | What is the Liberty Alliance?                                | 5  |
| 77  | 1.1.1 | The Liberty Vision   | 5  |
| 78  | 1.1.2 | The Liberty Mission  | 5  |
| 79  | 1.2   | What is Network Identity?                                    | 6  |
| 80  | 1.2.1 | The Liberty Objectives                                       | 6  |
| 81  | 2     | Liberty Version 1.0 User Experience Examples                 | 8  |
| 82  | 2.1   | Example of Identity Federation User Experience               | 8  |
| 83  | 2.2   | Example of Single Sign-on User Experience                    | 12 |
| 84  | 3     | Liberty Engineering Requirements Summary                     | 14 |
| 85  | 3.1   | General Requirements   | 14 |
| 86  | 3.1.1 | Client Device/User Agent Interoperability                    | 14 |
| 87  | 3.1.2 | Openness Requirements  | 14 |
| 88  | 3.2   | Functional Requirements                                      | 14 |
| 89  | 3.2.1 | Identity Federation  | 14 |
| 90  | 3.2.2 | Authentication   | 15 |
| 91  | 3.2.3 | Pseudonyms   | 15 |
| 92  | 3.2.4 | Global Logout  | 15 |
| 93  | 4     | Liberty Security Framework                                   | 15 |
| 94  | 5     | Liberty Architecture   | 17 |
| 95  | 5.1   | Web Redirection Architectural Component                      | 18 |
| 96  | 5.1.1 | HTTP-Redirect-Based Redirection                              | 19 |
| 97  | 5.1.2 | Form-POST-Based Redirection                                  | 20 |
| 98  | 5.1.3 | Cookies  | 20 |
| 99  | 5.1.4 | Web Redirection Summary                                      | 21 |
| 100 | 5.2   | Web Services Architectural Component                         | 21 |
| 101 | 5.3   | Metadata and Schemas Architectural Component                 | 21 |
| 102 | 5.4   | Single Sign-On and Identity Federation                       | 22 |
| 103 | 5.4.1 | Identity Federation  | 22 |
| 104 | 5.4.2 | Single Sign-on   | 26 |
| 105 | 5.4.3 | Profiles of the Single Sign-On and Federation Protocol       | 29 |
| 106 | 5.5   | Identity Provider Introduction                               | 32 |
| 107 | 5.6   | Single Logout  | 34 |
| 108 | 5.6.1 | Single Logout Profiles                                       | 35 |
| 109 | 5.7   | Example User Experience Scenarios                            | 36 |
| 110 | 5.7.1 | Scenario: Not Logged in Anywhere, No Common Domain Cookie    | 36 |
| 111 | 5.7.2 | Scenario: Not Logged in Anywhere, Has a Common Domain Cookie | 40 |
| 112 | 5.7.3 | Scenario: Logged in, Has a Common Domain Cookie              | 40 |
| 113 | 6     | References   | 40 |
| 114 |       |  |    |
| 115 |       |  |    |

## 115 1 Introduction

116 The Internet is now a prime vehicle for business, community, and personal interactions. The notion  
117 of *identity* is the crucial component of this vehicle. Today, one's identity on the Internet is  
118 fragmented across various identity providers — employers, Internal portals, various communities,  
119 and business services. This fragmentation yields isolated, high-friction, one-to-one customer-to-  
120 business relationships and experiences.

121  
122 *Federated network identity* is the key to reducing this friction and realizing new business taxonomies  
123 and opportunities, coupled with new economies of scale. In this new world of federated commerce, a  
124 user's online identity, personal profile, personalized online configurations, buying habits and history,  
125 and shopping preferences will be administered by the user and securely shared with the organizations  
126 of the user's choosing. A federated network identity model will ensure that critical private  
127 information is used by appropriate parties.

128  
129 The path to realizing a rich, fertile federated identity infrastructure can be taken in phases. The  
130 natural first phase is the establishment of a standardized, multivendor, Web-based single sign-on  
131 with simple federated identities based on today's commonly deployed technologies. This document  
132 presents an overview of the *Liberty Version 1.0 architecture*, which offers a viable approach for  
133 implementing such a single sign-on with federated identities. This overview first summarizes  
134 federated network identity, describes two key Liberty Version 1.0 user experience scenarios,  
135 summarizes the Liberty engineering requirements and security framework, and then provides a  
136 discussion of the Liberty Version 1.0 architecture.

137  
138 Policy/security and technical notes are highlighted throughout the document. Further details of the  
139 Liberty architecture are given in several technical documents associated with this overview,  
140 specifically [LibertyAuthnContext], [LibertyBindProf], [LibertyArchImpl], and  
141 [LibertyProtSchema]. Note: The more global term *Principal* is used for *user* in Liberty's technical  
142 documents. Definitions for Liberty-specific terms can be found in the [LibertyGloss]. Also, many  
143 abbreviations are used in this document without immediate definition because the authors believe  
144 these abbreviations are widely known, for example, HTTP and SSL. However, the definitions of  
145 these abbreviations can also be found in [LibertyGloss]. Note: Phrases and numbers in brackets [ ]  
146 refer to other documents; details of these references can be found in Section 6 (at the end of this  
147 document).

### 148 1.1 What is the Liberty Alliance?

149 The Liberty Alliance Project represents a broad spectrum of industries united to drive a new level of  
150 trust, commerce, and communications on the Internet.

#### 151 1.1.1 The Liberty Vision

152 The members of the Liberty Alliance envision a networked world across which individuals and  
153 businesses can engage in virtually any transaction without compromising the privacy and security of  
154 vital identity information.

#### 155 1.1.2 The Liberty Mission

156 To accomplish its vision, the Liberty Alliance will establish open technical specifications that  
157 support a broad range of network identity-based interactions and provide businesses with

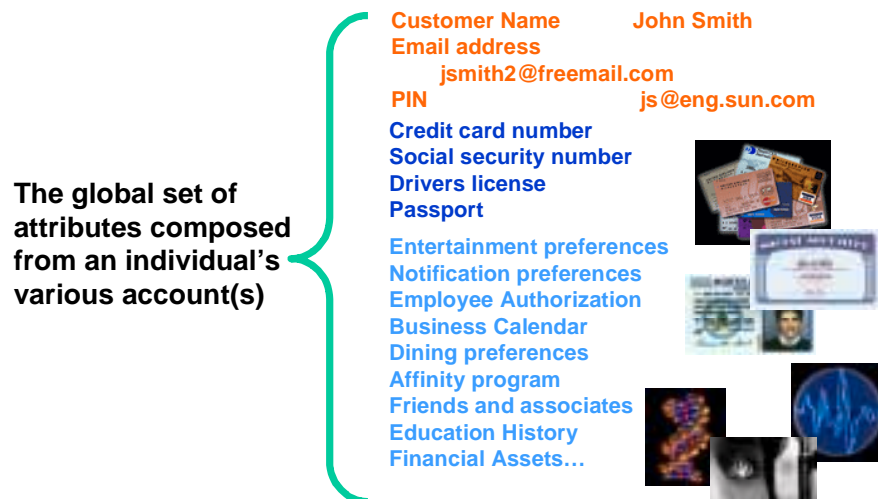
158  
159  
160  
161  
162  
163

- A basis for new revenue opportunities that economically leverage their relationships with consumers and business partners and
- A framework within which the businesses can provide consumers with choice, convenience, and control when using any device connected to the Internet.

## 164 1.2 What is Network Identity?

165 When users interact with services on the Internet, they often tailor the services in some way for their  
166 personal use. For example, a user may establish an account with a username and password and/or set  
167 some preferences for what information the user wants displayed and how the user wants it displayed.  
168 The network identity of each user is the overall global set of these attributes constituting the various  
169 accounts (see Figure 1).

### What is Network Identity?



170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184

**Figure 1: A network identity is the global set of attributes composed from a user's account(s).**

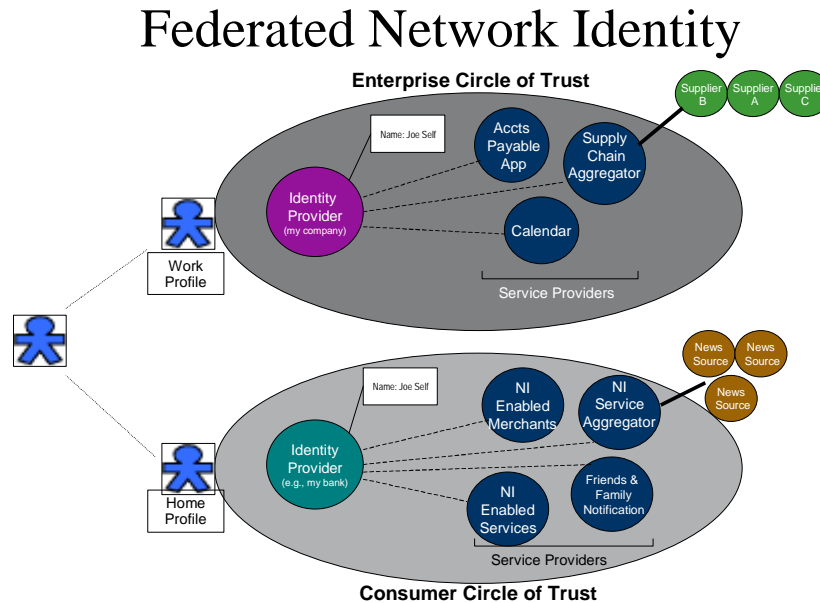
Today, users' accounts are scattered across isolated Internet sites. Thus the notion that a user could have a cohesive, tangible network identity is not realized.

### 1.2.1 The Liberty Objectives

The key objectives of the Liberty Alliance are to

- Enable consumers to protect the privacy and security of their network identity information
- Enable businesses to maintain and manage their customer relationships without third-party participation
- Provide an open single sign-on standard that includes decentralized authentication and authorization from multiple providers
- Create a network identity infrastructure that supports all current and emerging network access devices

185 These capabilities can be achieved when, first, businesses affiliate together into *circles of trust* based  
186 on Liberty-enabled technology and on operational agreements that define *trust relationships* between  
187 the businesses and, second, users federate the otherwise isolated accounts they have with these  
188 businesses (known as their *local identities*). In other words, a circle of trust is a federation of service  
189 providers and identity providers that have business relationships based on Liberty architecture and  
190 operational agreements and with whom users can transact business in a secure and apparently  
191 seamless environment. See Figure 2. Note: Operational agreement definitions are out of the scope of  
192 the Liberty Version 1.0 specifications.



193

194

Figure 2: Federated network identity and circles of trust

195

196 From a Liberty perspective, the salient actors in Figure 2 are the user, service providers, and identity  
197 providers.

198

199 Service providers are organizations offering Web-based services to users. This broad category  
200 includes practically any organization on the Web today, for example, Internet portals, retailers,  
201 transportation providers, financial institutions, entertainment companies, not-for-profit organizations,  
202 governmental agencies, etc.

203

204 Identity providers are service providers offering business incentives so that other service providers  
205 affiliate with them. Establishing such relationships creates the circles of trust shown in Figure 2. For  
206 example, in the enterprise circle of trust, the identity provider is a company leveraging employee  
207 network identities across the enterprise. Another example is the consumer circle of trust, where the  
208 user's bank has established business relationships with various other service providers allowing the  
209 user to wield his/her bank-based network identity with them. Note: A single organization may be  
210 both an identity provider and a service provider, either generally or for a given interaction.

211

212 These scenarios are enabled by service providers and identity providers deploying Liberty-enabled  
213 products in their infrastructure, but do not require users to use anything other than today's common  
214 Web browser.

## 2 Liberty Version 1.0 User Experience Examples

This section provides two simple, plausible examples of the Liberty Version 1.0 user experience, from the perspective of the user, to set the overall context for delving into technical details of the Liberty architecture in the Section 5. As such, actual technical details are hidden or simplified.

Note: the user experience examples presented in this section are non-normative and are presented for illustrative purposes only.

These user experience examples are based upon the following set of actors:

- Joe Self                      A user of Web-based online services.
- Airline.inc                  An airline maintaining an affinity group of partners. Airline.inc is an identity provider.
- CarRental.inc              A car rental company that is a member of the airline's affinity group. CarRental.inc is a service provider.

The Liberty Version 1.0 user experience has two main facets:

- Identity federation
- Single sign-on

Identity federation is based upon linking users' otherwise distinct service provider and identity provider accounts. This account linkage, or *identity federation*, in turn underlies and enables the other facets of the Liberty Version 1.0 user experience.

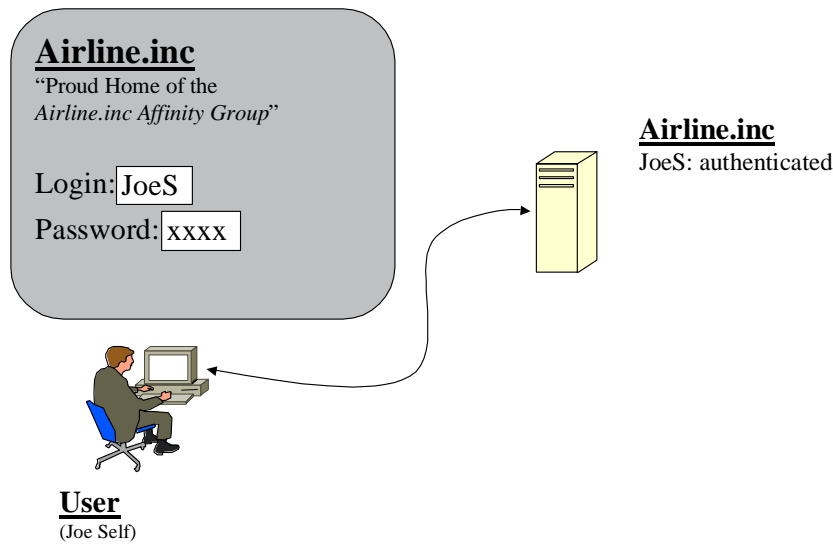
OVERALL POLICY/SECURITY NOTE: Identity federation must be predicated upon prior agreement between the identity and service providers. It should be additionally predicated upon providing notice to the user, obtaining the user's consent, and recording both the notice and consent in an auditable fashion. Providing an auditable record of notice and consent will enable both users and providers to confirm that notice and consent were provided and to document that the consent is bound to a particular interaction. Such documentation will increase consumer trust in online services. Implementors and deployers of Liberty-enabled technology should ensure that notice and user consent are auditably recorded in Liberty-enabled interactions with users, as appropriate.

Single sign-on enables users to sign on once with a member of a federated group of identity and service providers (or, from a provider's point of view, with a member of a circle of trust) and subsequently use various Websites among the group without signing on again.

### 2.1 Example of Identity Federation User Experience

The identity federation facet of the Liberty Version 1.0 user experience typically begins when Joe Self logs in to Airline.inc's Website, a Liberty-enabled identity provider, as illustrated in Figure 3.

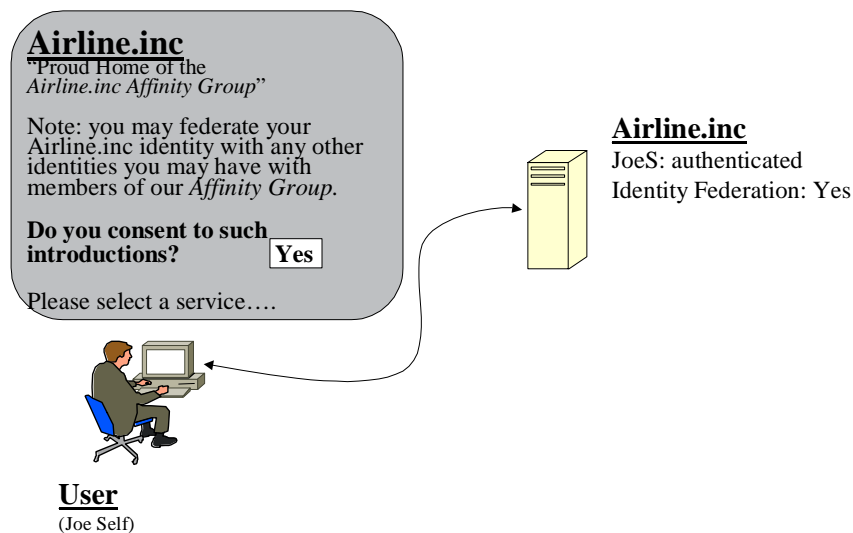
Note: Even though Joe Self is unaware of it, behind the scenes the identity provider is using Joe Self's credentials—his username and password in this case—to *authenticate* his identity. If successful, Joe Self is considered *authenticated*.



259  
260  
261  
262  
263  
264  
265  
266

**Figure 3: User logs in at a Liberty-enabled Website.**

Airline.inc. (as would any other identity provider that has created a circle of trust among its affinity group) will notify its eligible users of the possibility of federating their local identities among the members of the affinity group and will solicit permission to facilitate such introductions. See Figure 4.



267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277

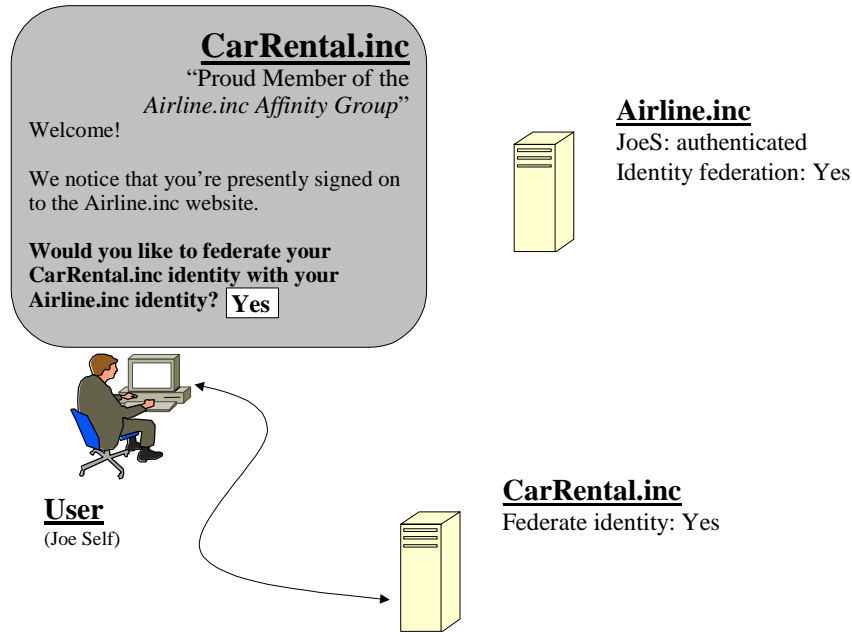
**Figure 4: User is notified of eligibility for identity federation and elects to allow introductions.**

**POLICY/SECURITY NOTE:** Figure 4 illustrates the user's consenting to introductions. An introduction is the means by which a service provider may discover which identity providers in the circle of trust have authenticated the user. Note: 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.

The act of introduction may be implemented via the Identity Provider Introduction Profile (as detailed in [LibertyBindProf]), or it may be implemented via other unspecified means, such as when the user agent is a Liberty-enabled client or proxy.

278  
279  
280  
281  
282  
283  
284  
285  
286

At some later point in time, typically minutes to a few hours, Joe Self may visit the Website of an affinity group member, for example, CarRental, Inc., whose site is CarRental.inc. Indeed, Joe Self may have followed an explicit link from the original Airline.inc Website to the CarRental.inc Website. In either case, CarRental.inc (a Liberty-enabled service provider) is able (because Joe Self elected to allow introductions) to discern that Joe Self is presently signed on to the Airline.inc Website and is able to present him with the opportunity to federate his local identities between CarRental.inc and Airline.inc. See Figure 5.

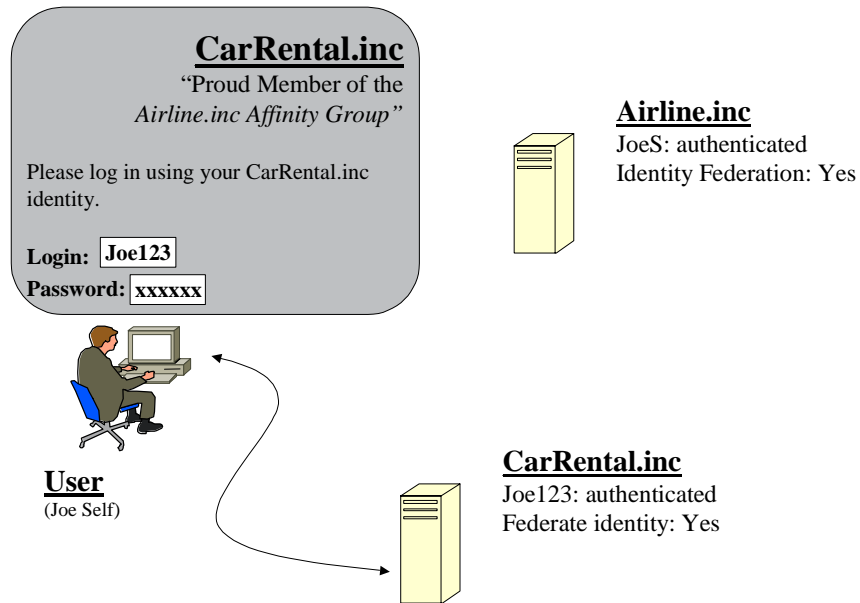


287  
288  
289  
290  
291  
292  
293

Figure 5: User is prompted to federate his local identities and selects “yes.”

Because Joe Self indicates that he wants to federate his local CarRental.inc and Airline.inc identities, he is asked to log in to the CarRental.inc Website using his local CarRental.inc identity. See Figure 6.





294

295

**Figure 6: User logs in using his local service provider identity.**

296

297

POLICY/SECURITY NOTE: As part of the federation activity, the service provider will authenticate the user. Whether the service provider asks for consent to federate the user's local identity before or after locally authenticating the user is a matter of local deployment policy.

298

299

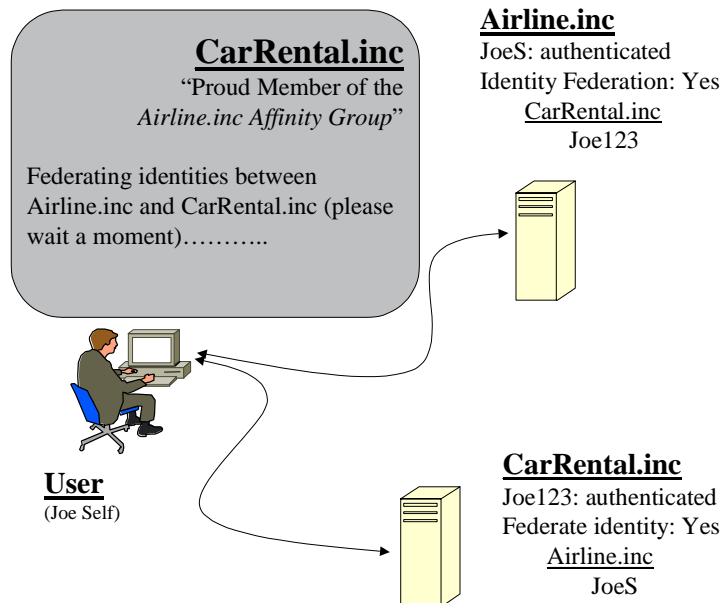
300

301

As a part of logging in to the CarRental.inc Website, Joe Self's local CarRental.inc identity is federated with his local Airline.inc identity. See Figure 7.

302

303



304

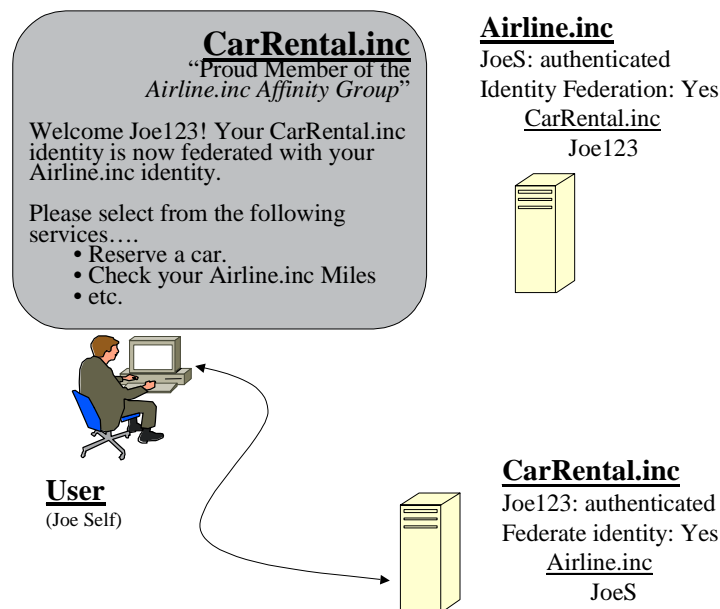
305

**Figure 7: The Websites federate the user's local identities.**

306  
307 Upon completion of the login and identity federation activity, Joe User is logged in to the  
308 CarRental.inc Website, and CarRental.inc delivers services to him as usual. In addition, the Website  
309 may now offer new selections because Joe Self’s local service provider (CarRental.inc) identity has  
310 been federated with his local identity provider (Airline.inc) identity. See Figure 8.

311  
312 TECHNICAL NOTE: Some figures illustrating the user experience, for example, Figure 7, show simplified,  
313 user-perspective notions of how identity federation is effected. In actuality, cleartext identifiers, for example,  
314 “JoeS” and “Joe123” WILL NOT be exchanged between the identity provider and service provider. Rather,  
315 opaque user handles will be exchanged. See 5.4.1 for details.

316  
317 Additionally, if errors are encountered in the process of authenticating and/or federating, the service provider  
318 will need to present appropriate indications to the user.  
319

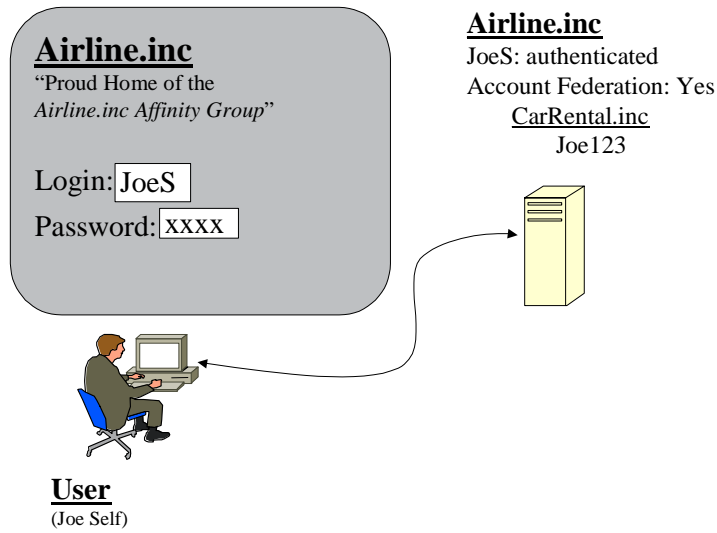


320  
321 **Figure 8: The service provider delivers services to user as usual.**

322  
323 POLICY/SECURITY NOTE: Business prerequisites must be met to offer identity federation. Two prerequisites  
324 are notifying the user of the capability to federate and soliciting consent to facilitate introductions. Another is  
325 creating agreements between the affinity group members to establish their policies for recognizing identities and  
326 honoring reciprocal authentication.

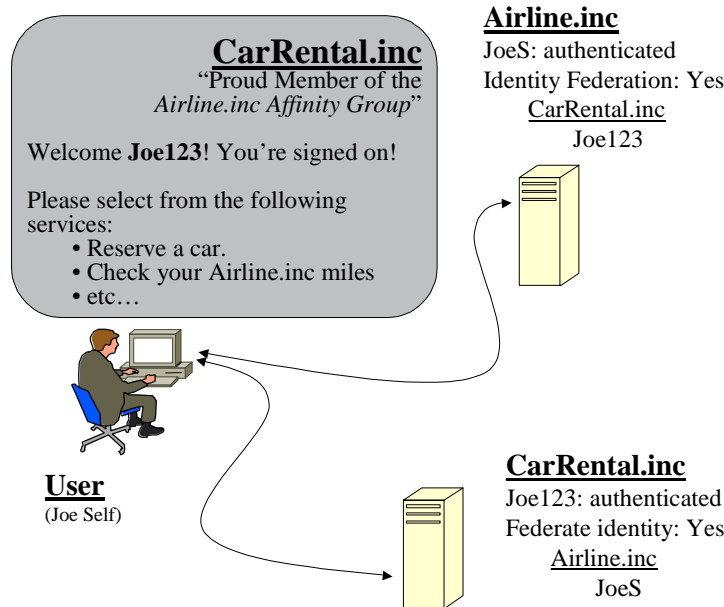
## 327 2.2 Example of Single Sign-on User Experience

328 Single sign-on builds upon identity federation and has a simple user experience. Joe Self logs in to  
329 the Airline.inc Website and later visits the CarRental.inc Website with which he has established  
330 identity federation. Joe Self’s authentication state with the Airline.inc Website is reciprocally  
331 honored by the CarRental.inc Website, and Joe Self is transparently logged in to the latter site. See  
332 Figure 9 and Figure 10.  
333



334  
335  
336  
337

Figure 9: User logs in to identity provider's Website using local identity.



338  
339  
340  
341  
342  
343  
344  
345  
346  
347

Figure 10: User proceeds to service provider's Website, and his authentication state is reciprocally honored by the service provider's Website.

A perceptive Joe Self will notice that his name in the CarRental.inc session is based upon his local CarRental.inc identity, rather than the local Airline.inc identity with which it has been federated.

**TECHNICAL NOTE:** Because users' actual account identifiers are not exchanged during federation, a service provider will not be able to display a user's identity provider identifier.

348 Also, many types of service provider Websites may not use a personally identifiable identifier in response to the  
349 user. For example, advertising-driven sites where users may specify display preferences, for example, a sporting  
350 events schedule site. The site may simply transparently refer to the user as “you,” for example, “Set your display  
351 preferences here...,” “Here is the list of upcoming events you’re interested in...,” etc.

352  
353 SECURITY/POLICY NOTE: Even though the user may be validly authenticated via the single sign-on  
354 mechanism, the user’s use of the service provider’s Website is still subject to local policy. For example, the site  
355 may have time-of-day usage restrictions, the site may be undergoing maintenance, the user’s relationship with  
356 the service provider may be in a particular state (for example, highly valued customer – show the user the bonus  
357 pages; troublesome customer – remind the user of unpaid bills and restrict some access).

### 358 **3 Liberty Engineering Requirements Summary**

359 This section summarizes the Liberty general and functional engineering requirements.

#### 360 **3.1 General Requirements**

361 The Liberty-enabled systems should follow the set of general principals outlined in 3.1.1 and 3.1.2.  
362 These principles cut across categories of functionality.

##### 363 **3.1.1 Client Device/User Agent Interoperability**

364 Liberty Version 1.0 clients encompass a broad range of presently deployed Web browsers, other  
365 presently deployed Web-enabled client access devices, and newly designed Web-enabled browsers  
366 or clients with specific Liberty-enabled features.

367  
368 The Liberty Version 1.0 architecture and protocol specifications must support a basic level of  
369 functionality across the range of Liberty Version 1.0 clients.

##### 370 **3.1.2 Openness Requirements**

371 The Liberty architecture and protocol specifications must provide the widest possible support for  
372

- 373 • Operating systems
- 374 • Programming languages
- 375 • Network infrastructures

376  
377 and must not impede multivendor interoperability between Liberty clients and services, including  
378 interoperability across circle of trust boundaries.

#### 379 **3.2 Functional Requirements**

380 The Liberty architecture and protocols must be specified so that Liberty-enabled implementations are  
381 capable of performing the following activities:

- 382
- 383 • Identity federation
- 384 • Authentication
- 385 • Use of pseudonyms
- 386 • Global logout

##### 387 **3.2.1 Identity Federation**

388 Requirements of identity federation stipulate that  
389

- 390
- Providers give the user notice upon identity federation and defederation.
- 391
- Service providers and identity providers notify each other about identity defederation.
- 392
- Each identity provider notifies appropriate service providers of user account terminations at
- 393
- the identity provider.
- 394
- Each service provider and/or identity provider gives each of its users a list of the user's
- 395
- federated identities at the identity provider or service provider.

### 396 3.2.2 Authentication

397 Authentication requirements include

398

- Supporting any method of navigation between identity providers and service providers on the part of the user, that is, how the user navigates from A to B (including click-through, favorites or bookmarks, URL address bar, etc.) must be supported.
- Giving the identity provider's authenticated identity to the user before the user gives credentials or any other personally identifiable information to the identity provider.
- Providing for the confidentiality, integrity, and authenticity of information exchanged between identity providers, service providers, and user agents, as well as mutually authenticating the identities of the identity providers and service providers, during the authentication and single sign-on processes.
- Supporting a range of authentication methods, extensibly identifying authentication methods, providing for coalescing authentication methods into authentication classes, and citing and exchanging authentication classes. Protocols for exchanging this information are out of the scope of the Liberty Version 1.0 specifications, however.
- Exchanging the following minimum set of authentication information with regard to a user: authentication status, instant, method, and pseudonym.
- Giving service providers the capability of causing the identity provider to reauthenticate the user using the same or a different authentication class. Programmatic exchange of the set of authentication classes for which a user is registered at an identity provider is out of the scope of the Liberty Version 1.0 specifications, however.

### 418 3.2.3 Pseudonyms

419 Liberty-enabled implementations must be able to support the use of pseudonyms that are unique on a  
420 per-identity-federation basis across all identity providers and service providers.

### 421 3.2.4 Global Logout

422 Liberty-enabled implementations must be able to support the notification of service providers when a  
423 user logs out at identity provider.

## 424 4 Liberty Security Framework

425 Table 1 generally summarizes the security mechanisms incorporated in the Liberty specifications,  
426 and thus in Liberty-enabled implementations, across two axes: channel security and message  
427 security. It also generally summarizes the security-oriented processing requirements placed on  
428 Liberty implementations. Note: This section is non-normative, please refer to [LibertyProtSchema]  
429 and [LibertyBindProf] for detailed normative statements regarding security mechanisms.  
430

431

**Table 1: Liberty security mechanisms**

| Security Mechanism         | Channel Security  | Message Security<br>(for Requests, Assertions) |
|----------------------------|---|--|
| Confidentiality            | Required  | Optional                                       |
| Per-message data integrity | Required  | Required                                       |
| Transaction integrity      | —   | Required                                       |
| Peer-entity authentication | Identity provider — Required<br>Service provider — Optional | —  |
| Data origin authentication | —   | Required                                       |
| Nonrepudiation             | —   | Required                                       |

432

433

434

435

436

437

438

439

Channel security addresses how communication between identity providers, service providers, and user agents is protected. Liberty implementations must use TLS1.0 or SSL3.0 for channel security, although other communication security protocols may also be employed, for example, IPsec, if their security characteristics are equivalent to TLS or SSL. Note: TLS, SSL, and equivalent protocols provide confidentiality and integrity protection to communications between parties as well as authentication.

440

441

Critical points of channel security include the following:

442

443

444

445

- In terms of authentication, service providers are required to authenticate identity providers using identity provider server-side certificates. Identity providers have the option to require authentication of service providers using service provider client-side certificates.

446

447

448

449

450

451

452

453

- Additionally, each service provider is required to be configured with a list of authorized identity providers, and each identity provider is required to be configured with a list of authorized service providers. Thus any service provider-identity provider pair must be mutually authorized before they will engage in Liberty interactions. Such authorization is in addition to authentication. (Note: The format of this configuration is a local matter and could, for example, be represented as lists of names or as sets of X.509 certificates of other circle of trust members).

454

455

456

- The authenticated identity of an identity provider must be presented to a user before the user presents personal authentication data to that identity provider.

457

458

459

460

Message security addresses security mechanisms applied to the discrete Liberty protocol messages passed between identity providers, service providers, and user agents. These messages are exchanged across the communication channels whose security characteristics were just discussed.

461

462

Critical points of message security include the following:

463

464

465

466

467

468

- Liberty protocol messages and some of their components are generally required to be digitally signed and verified. Signing and verifying messages provide data integrity, data origin authentication, and nonrepudiation. Therefore, identity providers and service providers are required to use key pairs that are distinct from the key pairs applied for TLS and SSL channel protection and that are suitable for long-term signatures.

469                    SECURITY/POLICY NOTE: Specifically, the <AuthnRequest> message of the Single Sign-  
470                    On and Federation Protocol defined in [LibertyProtSchema] may be signed or not signed as  
471                    specified by agreement between the identity provider and service provider and indicated by the  
472                    <AuthnRequestSigned> element of the provider metadata. Not signing this message may  
473                    be considered reasonable in some deployment contexts, for example, an enterprise network, where  
474                    access to the network and its systems is moderated by some means out of the scope of the Liberty  
475                    architecture.

- 476
- 477                    • In transactions between service providers and identity providers, requests are required to  
478                    be protected against replay, and received responses are required to be checked for correct  
479                    correspondence with issued requests. Time-based assurance of freshness may be  
480                    employed. These techniques provide transaction integrity.

481

482                    To become circle of trust members, providers are required to establish bilateral agreements on  
483                    selecting certificate authorities, obtaining X.509 credentials, establishing and managing trusted  
484                    public keys, and managing life cycles of corresponding credentials.

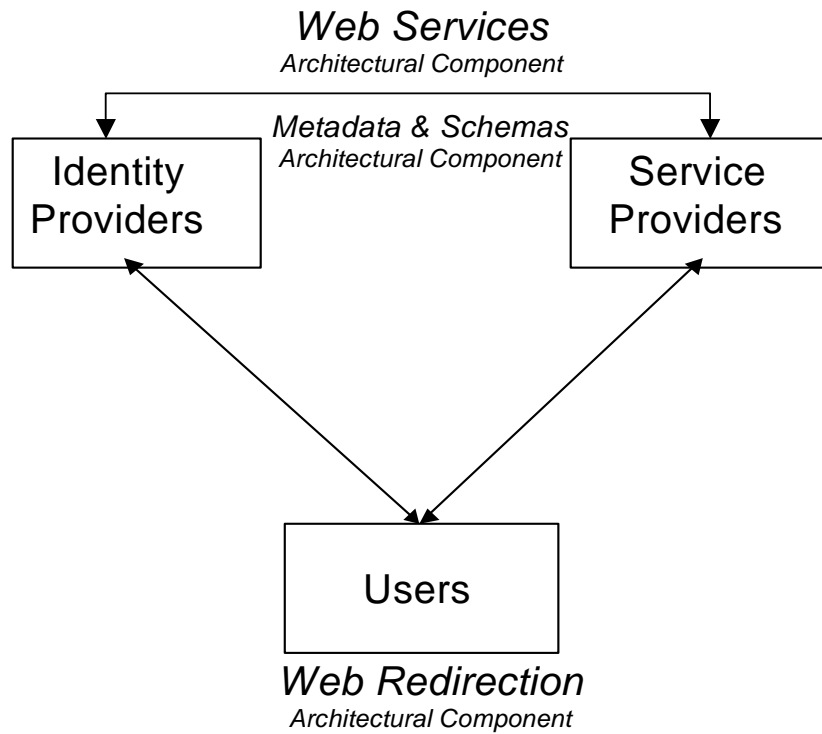
485

486                    SECURITY/POLICY NOTE: Many of the security mechanisms mentioned above, for example, SSL and TLS,  
487                    have dependencies upon, or interact with, other network services and/or facilities such as the DNS, time  
488                    services, firewalls, etc. These latter services and/or facilities have their own security considerations upon which  
489                    Liberty-enabled systems are thus dependent.

## 490                    **5 Liberty Architecture**

491                    The overall Liberty architecture is composed of three orthogonal architectural components (see  
492                    Figure 11):

- 493
- 494                    • Web redirection
  - 495                    • Web services
  - 496                    • Metadata and schemas



497  
498  
499  
500  
501  
502

**Figure 11: Overall Liberty architecture**

The role of each architectural component is summarized in Table 2:

**Table 2: Components of Liberty architecture**

|                      |  |
|----------------------|--|
| Web redirection      | Action that enables Liberty-enabled entities to provide services via today's user-agent-installed base.                            |
| Web services         | Protocol profiles that enable Liberty-enabled entities to directly communicate.  |
| Metadata and schemas | A common set of metadata and formats used by Liberty-enabled sites to communicate various provider-specific and other information. |

503  
504  
505  
506  
507  
508  
509  
510  
511

Sections 5.1 through 5.3 describe each architectural component. Sections 5.4 through 5.6 then relate the architectural components to the concrete protocols and profiles detailed in [LibertyProtSchema] and [LibertyBindProf], and 5.7 provides illustrations of user experience.

## 5.1 Web Redirection Architectural Component

The Web redirection architectural component is composed of two generic variants: HTTP-redirect-based redirection and form-POST-based redirection. Both variants create a communication channel between identity providers and service providers that is rooted in the user agent. See Figure 12.



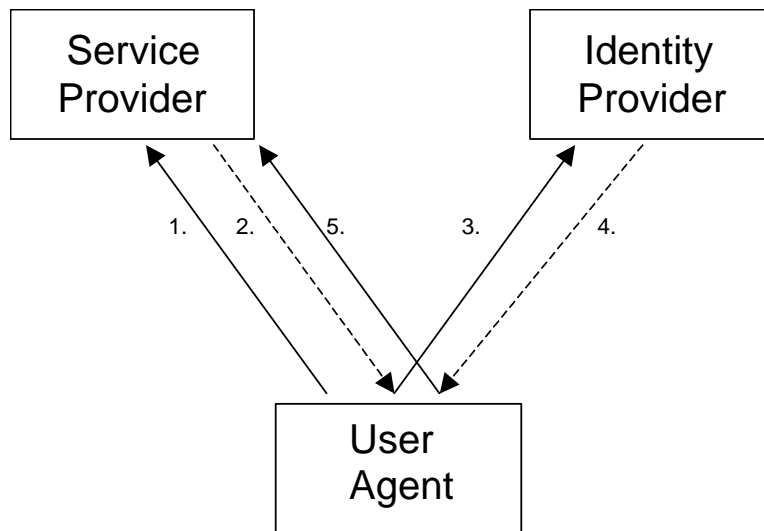


Figure 12: Web redirection between a service provider and an identity provider via the user agent

512  
513  
514

### 5.1.1 HTTP-Redirect-Based Redirection

HTTP-redirect-based redirection uses the HTTP redirection class of response (that is, *redirects*) of the HTTP protocol (see [RFC2616]) and the syntax of URIs (see [RFC1738] and [RFC2396]) to provide a communication channel between identity providers and service providers. Thus the steps shown in Figure 12 create a communication channel between the service provider and identity provider as follows:

521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539

1. The user agent sends an HTTP request to the service provider (typically a GET). In this step the user has typically clicked on a link in the Webpage presently displayed in the user agent.
2. The service provider responds with an HTTP response with a status code of 302 (that is, a redirect) and an alternate URI in the Location header field. In this example, the Location URI will point to the identity provider and will also contain a second, embedded URI pointing back to the service provider.
3. The user agent sends an HTTP request to the identity provider (typically a GET), specifying the complete URI taken from the Location field of the response returned in Step 2 as the argument of the GET. Note: This URI contains the second, embedded URI pointing back to the service provider.
4. The identity provider can then respond in kind with a redirect whose Location header field contains the URI pointing to the service provider (extracted from the GET argument URI supplied in Step 3) and optionally contains an embedded, second URI pointing back to itself.
5. The user agent sends an HTTP request to the service provider (typically a GET), specifying the complete URI taken from the Location field of the response returned in Step 4 as the argument of the GET. Note: This URI might contain any second, embedded URI pointing back to the identity provider.

Note: Both URIs are passed as arguments of HTTP GET requests, and the Location response-header field of redirect responses can contain either or both embedded URIs and other arbitrary data. Thus the identity provider and service provider can relatively freely exchange arbitrary information between themselves across this channel. See Table 3.

544

545

**Table 3: Embedding a parameter within an HTTP redirect**

|   |   |
|---|---|
| Location: <a href="http://www.foobar.com/auth">http://www.foobar.com/auth</a>                   | Redirects to foobar.com   |
| Location: <a href="http://www.foobar.com/auth?XYZ=1234">http://www.foobar.com/auth?XYZ=1234</a> | Redirects to foobar.com and also passes a parameter “XYZ” with the value “1234” |

546 **5.1.2 Form-POST-Based Redirection**

547 In form-POST-based redirection, the following steps in Figure 12 are modified as follows:

548

549       2. The service provider responds by returning an HTML form to the user agent containing an  
550 action parameter pointing to the identity provider and a method parameter with the value of POST.  
551 Arbitrary data may be included in other form fields. The form may also include a JavaScript or  
552 ECMAScript fragment that causes the next step to be performed without user interaction.

553       3. Either the user clicks on the Submit button, or the JavaScript or ECMAScript executes. In  
554 either case, the form and its arbitrary data contents are sent to the identity provider via the HTTP  
555 POST method.

556

557 The above process can be reversed in Steps 4 and 5 to effect form-POST-based communication in  
558 the opposite direction.

559 **5.1.3 Cookies**

560 POLICY/SECURITY NOTE: Use of cookies by implementors and deployers should be carefully considered,  
561 especially if a cookie contains either or both personally identifying information and authentication information.  
562 Cookies can be either ephemeral (that is, this session only) or persistent. Persistent cookies are of special  
563 concern because they are typically written to disk and persist across user agent invocations. Thus if a session  
564 authentication token is cached in a persistent cookie, the user exits the browser, and another person uses the  
565 system and relaunches the browser, then the second person could impersonate the user (unless any  
566 authentication time limits imposed by the authentication mechanism have expired).

567

568 Additionally, persistent cookies should be used *only* with the consent of the user. This consent step allows, for  
569 example, a user at a public machine to prohibit a persistent cookie that would otherwise remain in the user  
570 agent’s cookie cache after the user is finished.

571 **5.1.3.1 Why Not Use Cookies in General?**

572 Cookies are the HTTP state management mechanism specified in [RFC2965] and are a means for  
573 Web servers to store information, that is, *maintain state*, in the user agent. However, the default  
574 security setting in the predominant user agents allow cookies to be read only by the Website that  
575 wrote them. This discrimination is based on the DNS domains of the reading and writing sites.

576

577 To permit multiple identity providers and service providers in different DNS domains to  
578 communicate using cookies, users must lower the default security settings of their user agents. This  
579 option is often an unacceptable requirement.

580

581 Additionally, it is not uncommon for users and/or their organizations to operate their user agents with  
582 cookies turned off.

583 **5.1.3.2 Where Cookies are Used**

584 In the Liberty context, cookies might be used for maintaining local session state, and cookies are  
585 used in addressing the introduction problem (see 5.5).

586

587 The fact that identity providers cannot arbitrarily send data to service providers via cookies does not  
588 preclude identity providers and service providers from writing cookies to store local session state and  
589 other, perhaps persistent, information.

#### 590 **5.1.4 Web Redirection Summary**

591 Web redirection is not an ideal distributed systems architecture.

592

593 POLICY/SECURITY NOTE: Communications across Web redirection channels as described in 5.1.1 through  
594 5.1.3 have many well-documented security vulnerabilities, which should be given careful consideration when  
595 designing protocols utilizing Web redirection. Such consideration was incorporated into the design of the  
596 profiles specified in [LibertyBindProf], and specific considerations are called out as appropriate in that  
597 document (for example, regarding cleartext transmissions and caching vulnerabilities). Examples of security  
598 vulnerabilities include

599

- 600 • **Interception**: Such communications go across the wire in cleartext unless all the steps in 5.1.1 through  
601 5.1.3 are carried out over an SSL or TLS session or across another secured communication transport, for  
602 example, an IPsec-based VPN.
- 603 • **User agent leakage**: Because the channel is redirected through the user agent, many opportunities arise for  
604 the information to be cached in the user agent and revealed later. This caching is possible even if a secure  
605 transport is used because the conveyed information is kept in the clear in the browser. Thus any sensitive  
606 information conveyed in this fashion needs to be encrypted on its own before being sent across the channel.

607

608 TECHNICAL NOTE: A key limitation of Web redirection is the overall size of URIs passed as arguments of  
609 GET requests and as values of the Location field in redirects. These elements have size limitations that vary  
610 from browser to browser and are particularly small in some mobile handsets. These limitations were  
611 incorporated into the design of the protocols specified in [LibertyProtSchema] and [LibertyBindProf].

612

613 In spite of the vulnerabilities and limitations of Web redirection, use of this mechanism enables  
614 distributed, cross-domain interactions, such as single sign-on, with today's deployed HTTP  
615 infrastructure on the Internet.

616

617 Both generic variants of Web redirection underlie several of the profiles specified in  
618 [LibertyBindProf]: Single Sign-On and Federation, Identity Federation Termination Notification,  
619 Identity Provider Introduction, and Single Logout.

#### 620 **5.2 Web Services Architectural Component**

621 Various Liberty protocol interaction steps are profiled to occur directly between system entities in  
622 addition to other steps occurring via Web redirection and are based on RPC-like protocol messages  
623 conveyed via SOAP (see [SOAP1.1]). SOAP is a widely implemented specification for RPC-like  
624 interactions and message communications using XML and HTTP and hence is a natural fit for this  
625 architectural component.

#### 626 **5.3 Metadata and Schemas Architectural Component**

627 *Metadata and schemas* is an umbrella term generically referring to various subclasses of information  
628 and their formats exchanged between service providers and identity providers, whether via protocol  
629 or out of band. The subclasses of exchanged information are

630

- 631 • **Account/Identity**: In Liberty Version 1.0, account/identity is simply the opaque user handle  
632 that serves as the name that the service provider and the identity provider use in referring to

633 the user when communicating. In future Liberty phases, it will encompass various attributes.  
634

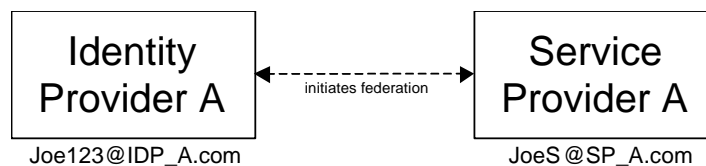
- 635 • **Authentication Context:** Liberty explicitly accommodates identity provider use of arbitrary  
636 authentication mechanisms and technologies. Different identity providers will choose  
637 different technologies, follow different processes, and be bound by different legal obligations  
638 with respect to how they authenticate users. The choices that an identity provider makes here  
639 will be driven in large part by the requirements of the service providers with which the  
640 identity provider has federated. Those requirements, in turn, will be determined by the nature  
641 of the service (that is, the sensitivity of any information exchanged, the associated financial  
642 value, the service providers risk tolerance, etc) that the service provider will be providing to  
643 the user. Consequently, for anything other than trivial services, if the service provider is to  
644 place sufficient confidence in the authentication assertions it receives from an identity  
645 provider, the service provider must know which technologies, protocols, and processes were  
646 used or followed for the original authentication mechanism on which the authentication  
647 assertion is based. The authentication context schema provides a means for service providers  
648 and identity providers to communicate such information (see [LibertyAuthnContext]).  
649
- 650 • **Provider Metadata:** For identity providers and service providers to communicate with each  
651 other, they must a priori have obtained metadata regarding each other. These provider  
652 metadata include items such as X.509 certificates and service endpoints. [LibertyProtSchema]  
653 defines metadata schemas for identity providers and service providers that may be used for  
654 provider metadata exchange. However, provider metadata exchange protocols are outside the  
655 scope of the Liberty Version 1.0 specifications.

## 656 5.4 Single Sign-On and Identity Federation

657 The single sign-on and identity federation aspects of Liberty are facilitated by the Single Sign-On  
658 and Federation Protocol, which is specified in [LibertyProtSchema]. It facilitates both identity  
659 federation (see 5.4.1) and single sign-on (see 5.4.2) in a single overall protocol flow. The various  
660 profiles of the overall protocol flow that are defined in [LibertyBindProf] are discussed in 5.4.3.

### 661 5.4.1 Identity Federation

662 The first time that users use an identity provider to log in to a service provider they must be given the  
663 option of federating an existing local identity on the service provider with the identity provider login  
664 to preserve existing information under the single sign-on. See Figure 13. It is critical that, in a system  
665 with multiple identity providers and service providers, a mechanism exists by which users can be (at  
666 their discretion) uniquely identified across the providers. However, it is technically challenging to  
667 create a globally unique ID that is not tied to a particular identity provider and a business challenge  
668 to ensure the portability of globally unique IDs.  
669



670  
671  
672

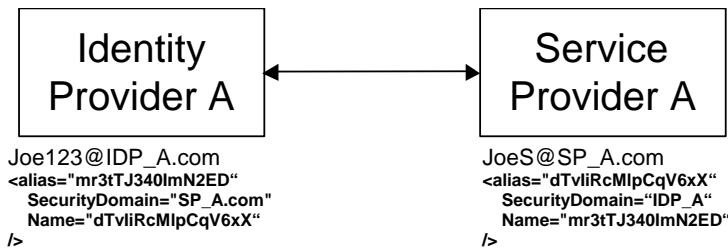
Figure 13: User initiates federation of two identities

673 An explicit trust relationship, or chain, is created with the opt-in identity federation that occurs the  
 674 first time a user logs in to a service provider using an identity provider. While multiple identities can  
 675 be federated to each other, an explicit link exists between each identity. Providers cannot skip over  
 676 each other in the trust chain to request information on or services for a user because user identity  
 677 information must be checked at each step. Therefore, the only requirement is that, when two  
 678 elements of a trust chain communicate, they can differentiate users.

679  
 680 Members of the circle of trust are not required to provide the actual account identifier for a user and  
 681 can instead provide a handle for a particular user. Members can also choose to create multiple  
 682 handles for a particular user. However, identity providers must create a single handle for each service  
 683 provider that has multiple Websites so that the handle can be resolved across the Websites.

684  
 685 Because both the identity provider and service provider in such a federation need to remember the  
 686 other's handle for the user, they create entries in their user directories for each other and note each  
 687 other's handle for the user. See Figure 14 and Figure 15.

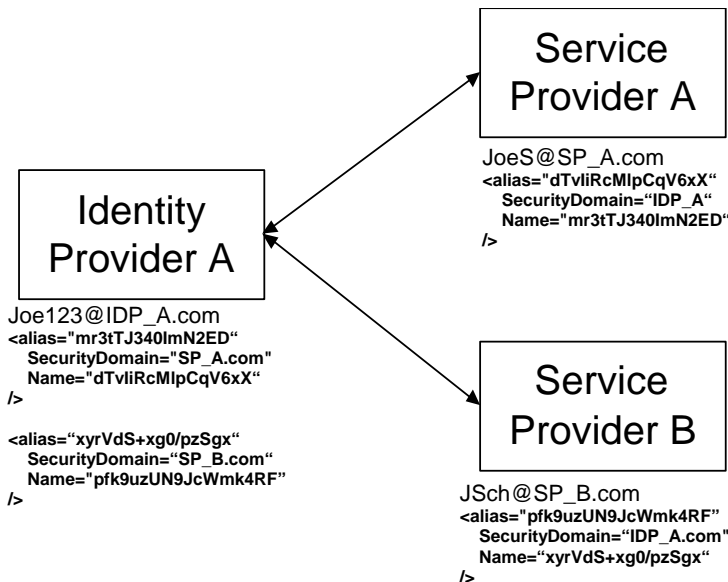
688



689

690 **Figure 14: User directories of the identity provider and service provider upon identity federation**

691



692

693 **Figure 15: User directories of the identity provider and multiple service providers**  
 694 **upon identity federation**

695

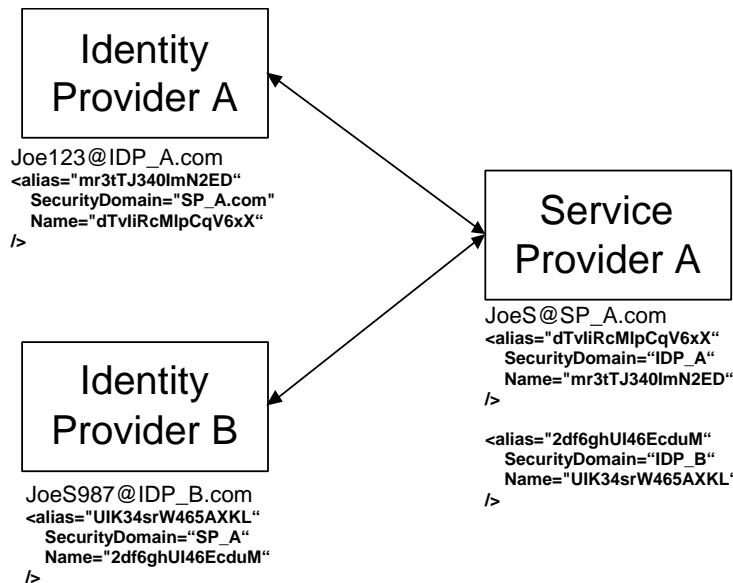
696

697

POLICY/SECURITY NOTE:

- 698 1. Observe in Figure 15 that SP\_A and SP\_B cannot communicate directly about Joe Self. They can only  
699 communicate with the identity provider individually. This feature is desirable from policy and security  
700 perspectives. If Joe Self wishes the service providers to be able to exchange information about him,  
701 then he must explicitly federate the two service provider identities, effectively opting in.  
702
- 703 Another aspect of this feature is that if the user's local identity is compromised on, for example, SP\_A,  
704 the local identities at IDP\_A or SP\_B are not necessarily also compromised.  
705
- 706 2. Properties of the user handles, for example, `mr3tTJ340ImN2ED`, (also known as *name identifiers*) need to  
707 be carefully considered. It may not be enough for them to be opaque. Considerations of the  
708 construction of name identifiers are discussed in [LibProtSchema]. Additionally, user handles should  
709 be refreshed periodically. Service providers may refresh the user handles they optionally supply to  
710 identity providers via the register name identifier profile defined in [LibertyBindProf]. Liberty Version  
711 1.0 has no provision for an automated refresh mechanism for name identifiers issued by identity  
712 providers.  
713

714 While it is obvious that a user can sign in at multiple service providers with an identity provider, a  
715 user can also link multiple identity providers to a particular service provider. See Figure 16. This  
716 ability proves useful when a user switches from a work computer to a home computer or from a  
717 computer to a mobile device, each of which may be associated with a different identity provider and  
718 circle of trust.  
719

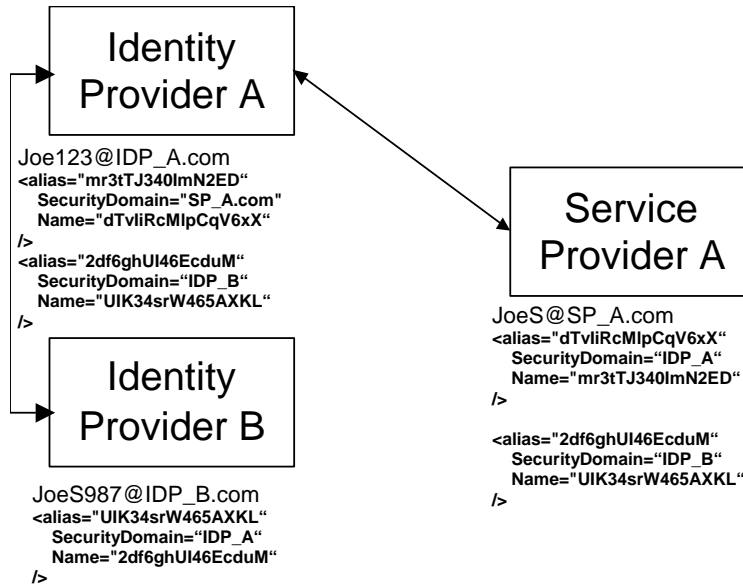


720  
721 **Figure 16: A user with two identity providers federated to a service provider**

722 **POLICY/SECURITY NOTE:** Subtle considerations arise here in terms of how easy it is for a user to switch  
723 between identities and how this capability is materialized. IDP\_A may belong to the same circles of trust as  
724 more than one of the user's devices. Therefore, certain questions arise, for example, How do users know to  
725 which (or both) identity provider they are presently logged in? Features satisfying such questions are a way for  
726 identity providers and circles of trust to differentiate themselves.  
727

728  
729 While federating two identity providers to a service provider enables the user to log in to the service  
730 provider using either identity provider, the user must remember to federate new service providers to  
731 both identity providers, which can be a cumbersome process. An alternative is for the user to federate  
732 identity providers together and set policies enabling identity providers to access each other's

733 information. See Figure 17. The user can then use a preferred identity provider to log in to service  
734 providers, but always has the choice of adding additional identity providers to a service provider.  
735



736  
737 **Figure 17: A user with two identity providers federated**

738  
739 POLICY/SECURITY NOTE:

- 740  
741 1. The semantics of such a federated relationship between identity providers are not dictated by the  
742 underlying Liberty protocols. These semantics will need to be addressed by the agreements between the  
743 identity providers and the capabilities of the deployed Liberty-enabled implementations.

744  
745 For example, if the Liberty protocols enable it and if federations can be either bidirectional or  
746 unidirectional, these capabilities will need to be addressed in the circle of trust business agreements and  
747 in the user interface.

- 748  
749 2. How are federation failures materialized to the user? Circle of trust policies should address how  
750 federation failures are materialized to users.  
751  
752 3. Appropriate portions of the assertions and contracts passed between the identity provider and the  
753 service provider to effect federation should be logged.  
754  
755 4. By creating many local identities with many service providers and/or identity providers and then  
756 federating them, users possess many sets of local credentials that may be used as a basis to authenticate  
757 with many service providers via single sign-on. This situation constitutes a risk. For example, every  
758 identity provider that possesses reusable user credentials, for example, a username and password, can  
759 impersonate the user at every service provider federated with that account.

760  
761 In the normal course of events, some local credentials may go unused for periods of time because the  
762 user is making use of the local account via single sign-on from another identity provider. Thus a means  
763 of controlling the growth of a user's set of local credentials might be to offer the user the option of  
764 invalidating local credentials at identity federation time and also perhaps after a certain number of  
765 times of visiting the Website without using them.

766 **5.4.1.1 No Need for Global Account/Identity Namespace**

767 Given the above architecture where users opt to federate identities at different identity providers and  
768 service providers, a global namespace across all of the players should not be needed. Circle of trust  
769 members can communicate with each other, about or on a user's behalf, only when a user has created  
770 a specific federation between the local identities and has set policies for that federation. Although  
771 long chains of identity providers and service providers can be created, the user's identity is federated  
772 in each link in the chain and, therefore, a globally unique ID need not exist for that user across all of  
773 the elements of the chain. See Figure 17.

774 **5.4.1.2 Federation Management: Defederation**

775 Users will have the ability to terminate federations, or *defederate identities*. [LibertyProtSchema] and  
776 [LibertyBindProf] specify a Federation Termination Notification Protocol and related profiles. Using  
777 this protocol, a service provider may initiate defederation with an identity provider or vice versa. The  
778 nominal user experience is for the user to select a Defederate link on a service provider's or identity  
779 provider's Webpage. This link initiates defederation with respect to some other, specific, identity  
780 provider or service provider.

781  
782 When defederation is initiated at an identity provider, the identity provider is stating to the service  
783 provider that it will no longer provide user identity information to the service provider and that the  
784 identity provider will no longer respond to any requests by the service provider on behalf of the user.  
785

786 When defederation is initiated at a service provider, the service provider is stating to the identity  
787 provider that the user has requested that the identity provider no longer provide the user identity  
788 information to the service provider and that service provider will no longer ask the identity provider  
789 to do anything on the behalf of the user.

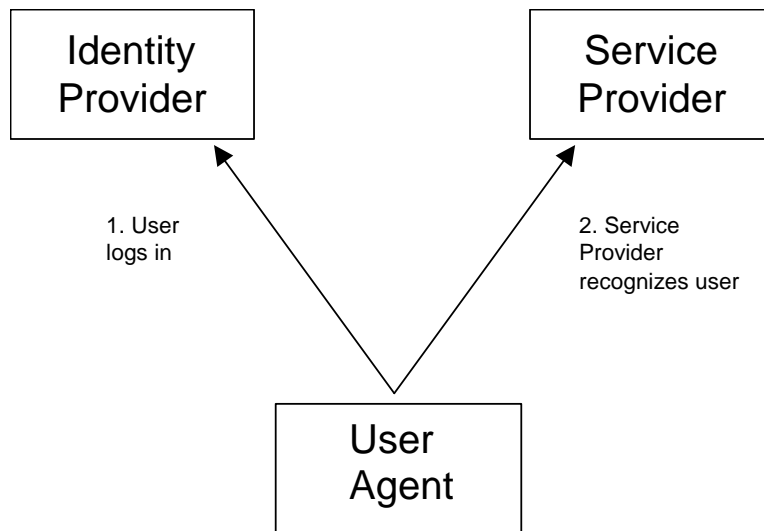
790  
791 POLICY/SECURITY NOTE: Regarding defederation, several issues must be considered:  
792

- 793 • The user should be authenticated by the provider at which identity defederation is being initiated.
- 794
- 795 • Providers should ask the user for confirmation before performing defederation and appropriately log  
796 the event and appropriate portions of the user's authentication information.
- 797
- 798 • Other means of federation termination are possible, such as federation expiration and termination of  
799 business agreements between service providers and identity providers.

800 **5.4.2 Single Sign-on**

801 Single sign-on is enabled once a user's identity provider and service provider identities are federated.  
802 From a user's perspective, single sign-on is realized when the user logs in to an identity provider and  
803 uses multiple affiliated service providers without having to sign on again (see Figure 18). This  
804 convenience is accomplished by having federated the user's local identities between the applicable  
805 identity providers and the service providers. The basic user single sign-on experience is illustrated in  
806 the 5.4.1.  
807





808  
809 **Figure 18: User logs in at identity provider and is recognized by service provider**

810  
811 [LibertyBindProf] specifies single sign-on by profiling both the “Browser/Artifact Profile” and the  
812 “Browser/Post Profile” of SAML (see [SAMLBind]).

813  
814 POLICY/SECURITY NOTE: Regarding authentication, single sign-on, credentials, etc., several issues must be  
815 considered:

816  
817 **Authentication Mechanisms are Orthogonal to Single Sign-On**

818  
819 Single sign-on is a means by which a service provider or identity provider may convey to another service  
820 provider or identity provider that the user is in fact authenticated. The means by which the user was originally  
821 authenticated is called the authentication mechanism. Examples of authentication mechanisms are username  
822 with password (*not* HTTP Basic Auth), certificate-based (for example, via SSL or TLS), Kerberos, etc.

823  
824 **Credentials**

825  
826 Credentials are relied upon in a number of ways in a single sign-on system and are often the basis for  
827 establishing trust with the credential bearer. Credentials may represent security-related attributes of the bearer,  
828 including the owner’s identity. Sensitive credentials that require special protection, such as private  
829 cryptographic keys, must be protected from unauthorized exposure. Some credentials are intended to be shared,  
830 such as public-key certificates.

831  
832 Credentials are a general notion of the data necessary to prove an assertion. For example, in a password-based  
833 authentication system, the user name and password would be considered credentials. However, the use of  
834 credentials is not limited to authentication. Credentials may also be relied upon in the course of making an  
835 authorization decision.

836  
837 As mentioned above, certain credentials must be kept confidential. However, some credentials not only need to  
838 remain confidential, but also must be integrity-protected to prevent them from being tampered with or even  
839 fabricated. Other credentials, such as the artifacts described in 5.4.3.1, must have the properties of a nonce. A  
840 nonce is a random or nonrepeating value that is included in data exchanged by a protocol, usually for  
841 guaranteeing liveness and thus detecting and protecting against replay attacks.

842  
843 **Authentication Type, Multitiered Authentication**

844  
845 All authentication assertions should include an authentication type that indicates the quality of the credentials  
846 and the mechanism used to vet them. Credentials used to authenticate a user or supplied to authorize a

847 transaction and/or the authentication mechanism used to vet the credentials may not be of sufficient quality to  
848 complete the transaction.  
849

850 For example, a user initially authenticates to the identity provider using username and password. The user then  
851 attempts to conduct a transaction, for instance, a bank withdrawal, which requires a stronger form of  
852 authentication. In this case the user must present a stronger assertion of identity, such as a public-key certificate  
853 or something ancillary such as birthdate, mother's maiden name, etc. This act is *reauthentication* and the overall  
854 functionality is *multitiered authentication*. Wielding multitiered authentication can be a policy decision at the  
855 service provider and can be at the discretion of the service provider. Or it might be established as part of the  
856 contractual arrangements of the circle of trust. In this case, the circle of trust members can agree among  
857 themselves upon the trust they put in different authentication types and of each other's authentication assertions.  
858 Such an agreement's form may be similar to today's certificate practice statements (CPS) (for example, see  
859 <http://www.verisign.com/repository/cps20/cps20.pdf>). The information cited in such a document may include  
860

- 861 • User identification methods during credentials enrollment
- 862 • Credentials renewal frequency
- 863 • Methods for storing and protecting credentials (for example, smartcard, phone, encrypted file on hard  
864 drive, etc.)  
865

866 Note: While the current Liberty specifications allow service providers, identity providers, and user agents to  
867 support authentication using a range of methods, the methods and their associated protocol exchanges are  
868 not specified within Liberty documents. Further, the scope of the current Liberty specifications does not  
869 include a means for a communicating identity provider and user agent to identify a set of methods that they  
870 are both equipped to support. As a result, support for the Liberty specifications is not in itself sufficient to  
871 ensure effective interoperability between arbitrary identity providers and user agents using arbitrary methods  
872 and must, instead, be complemented with data obtained from other sources.  
873

874 Also, the scope of the current Liberty specifications does not include a means for a service provider to  
875 interrogate an identity provider and determine the set of authentication profiles for which a user is registered  
876 at that identity provider. As a result, effective service provider selection of specific profiles to authenticate a  
877 particular user will require access to out-of-band information describing users' capabilities.  
878

879 For example, members of a given circle of trust may agree that they will label an authentication assertion based  
880 on PKI technology and face-to-face user identity verification with substantiating documentation at enrollment  
881 time to be of type "Strong." Then, when an identity provider implementing these policies and procedures asserts  
882 that a user has logged in using the specified PKI-based authentication mechanism, service providers rely upon  
883 said assertion to a certain degree. This degree of reliance is likely different from the degree put into an assertion  
884 by an identity provider who uses the same PKI-based authentication mechanism, but who does not claim to  
885 subject the user to the same amount of scrutiny at enrollment time.  
886

887 This issue has another dimension: Who performs the reauthentication? An identity provider or the service  
888 provider itself? This question is both an implementation and deployment issue and an operational policy issue.  
889 Implementations and deployments need to support having either the identity provider or the service provider  
890 perform reauthentication when the business considerations dictate it (that is, the operational policy). For  
891 example, a circle of trust may decide that the risk factors are too large for having the identity provider perform  
892 reauthentication in certain high-value interactions and that the service provider taking on the risk of the  
893 interaction must be able to perform the reauthentication.  
894

### 895 **Mutual Authentication**

896

897 Another dimension of the authentication type and quality space is mutual authentication. For a user  
898 authenticating himself to an identity provider, mutual authentication implies that the identity provider server  
899 authenticates itself with the user as well as vice versa. Mutual authentication is a function of the particular  
900 authentication mechanism employed. For example, any user authentication performed over SSL or TLS is  
901 mutual authentication because the server is authenticated to the client by default with SSL or TLS. This feature  
902 can be the basis of some greater assurance, but does have its set of vulnerabilities. The server may be wielding a  
903 bogus certificate, and the user may not adequately inspect it or understand the significance.  
904

905 **Validating Liveness**

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

*Liveness* refers to whether the user who authenticated at time  $t_0$  is the same user who is about to perform a given operation at time  $t_1$ . For example, a user may log in and perform various operations and then attempt to perform a given operation that the service provider considers high-value. The service provider may initiate reauthentication to attempt to validate that the user operating the system is still the same user that authenticated originally. Even though such an approach has many vulnerabilities, that is, it fails completely in the case of a rogue user, it does at least augment the service provider's audit trail. Therefore, at least some service providers will want to do it.

Authentication assertions from identity providers contain a `<ReauthenticationOnOrAfter>` element. If this attribute was specified and the time of the user request is past the specified reauthentication time, the service provider should redirect the user back to the identity provider for reauthentication.

919 **Communication Security**

920

921

922

923

924

A service provider can reject communications with an identity provider for various reasons. For example, it may be the policy of a service provider to require that all protocol exchanges between it and the bearer of a credential commence over a communication protocol that has certain qualities such as bilateral authentication, integrity protection, and message confidentiality.

925 **5.4.3 Profiles of the Single Sign-On and Federation Protocol**

926

927

928

929

930

931

932

933

934

935

936

937

938

The Single Sign-On and Federation Protocol, as specified in [LibertyProtSchema], defines messages exchanged between service providers and identity providers. The concrete mapping of these messages to particular transfer (for example, HTTP) and/or messaging (for example, SOAP) protocols and precise protocol flows are specified in [LibertyBindProf]. These mappings are called *profiles*. The Single Sign-On and Federation Protocol specifies four profiles. The following sections summarize each profile. For a detailed discussion of the common interactions and processing rules of these profiles and for details about each profile, see [LibertyBindProf].

**TECHNICAL NOTE:** The Single Sign-On and Federation Protocol and related profiles specify means by which service providers indicate to identity providers the particular profile they wish to employ. The primary means is the `<lib:ProtocolProfile>` element of the `<lib:AuthnRequest>` message, which is employed by all profiles of the Single Sign-On and Federation Protocol. Note: The Liberty-enabled client and proxy profile employs additional means.

939 **5.4.3.1 Liberty Browser Artifact Profile**

940

941

942

943

944

945

946

947

The Liberty browser artifact profile specifies embedding an artifact in a URI exchanged between the identity provider and service provider via Web redirection and also requires direct communication between the service provider and the identity provider. The artifact itself is an opaque user handle with which the service provider can query the identity provider to receive a full SAML assertion. The motivation for this approach is that the artifact can be small enough in its URI-encoded form to fit in a URI without concern for size limitations. The artifact has the property of being an opaque, pseudo-random nonce that can be used only once. These properties are countermeasures against replay attacks. The randomness property protects the artifact from being guessed by an adversary.

948 **5.4.3.2 Liberty Browser POST Profile**

949

950

951

952

953

Modern browsers that support JavaScript or ECMAScript can perform the redirect by sending an HTML page with form elements that contain data with a JavaScript or ECMAScript that automatically posts the form. Legacy browsers, or browsers with scripting disabled, must embed the data within the URI.

954 The Liberty browser POST profile embeds an assertion within an HTTP form per the form-POST-  
955 based redirection (see 5.1.2). As a result, this profile does not require any direct communication  
956 between the service provider and the identity provider to obtain an assertion. Because the size  
957 limitation is greater when using HTML forms than URLs, a full authentication assertion can be  
958 included. See Figure 19.

```
959  
960 <HTML>  
961 <BODY ONLOAD=" javascript:document.forms[0].submit()">  
962 <FORM METHOD="POST" ACTION="www.foobar.com/auth">  
963 <INPUT TYPE="HIDDEN" NAME="FOO" VALUE="1234"/>  
964 </FORM>  
965 </BODY>  
966 </HTML>
```

967 **Figure 19: Example of JavaScript-based HTML form autosubmission with hidden fields**

968  
969 TECHNICAL NOTE: It must be stressed that Liberty browser POST profile should be supported only in  
970 addition to Liberty browser artifact profile due to its dependence on JavaScript (or ECMAScript).

971  
972 POLICY/SECURITY NOTE: Implementors and deployers should provide for logging appropriate portions of  
973 the authentication assertion.

### 974 5.4.3.3 Liberty WML POST Profile

975 The Liberty WML POST profile relies on the use of WML events to instruct a WML browser to  
976 submit a HTTP form. WML browsers are typical on mobile handsets. The browsers on such handsets  
977 communicate via a dedicated proxy, a WAP gateway. This proxy converts the Wireless Session  
978 Protocol of the handset into HTTP. Note: The service provider and identity provider will be  
979 contacted using only HTTP.

980  
981 TECHNICAL NOTE: The primary difference between this profile and the Liberty browser POST profile is that  
982 certain responses from the service provider and identity provider to the user agent contain WML rather than  
983 HTML.

984  
985 The difference between this profile and the Liberty-enabled client and proxy profile is that this profile is  
986 designed to accommodate standard, unmodified WML browsers, while the Liberty-enabled client and proxy  
987 profile assumes a browser and/or proxy with built-in Liberty protocol capabilities.

### 988 5.4.3.4 Liberty-Enabled Client and Proxy Profile

989 The Liberty-enabled client and proxy profile specifies interactions between Liberty-enabled clients  
990 and/or proxies, service providers, and identity providers. A Liberty-enabled client is a client that has,  
991 or knows how to obtain, knowledge about the identity provider that the user wishes to use with the  
992 service provider. In addition a Liberty-enabled client receives and sends Liberty messages in the  
993 body of HTTP requests and responses using POST, rather than relying upon HTTP redirects and  
994 encoding protocol parameters into URLs. Therefore, Liberty-enabled clients have no restrictions on  
995 the size of the Liberty protocol messages.

996  
997 A Liberty-enabled proxy is a HTTP proxy (typically a WAP gateway) that emulates a Liberty-  
998 enabled client.

999  
1000 TECHNICAL NOTE: The differences between this profile and the other Liberty POST-based profiles are that

- 1001 • It does not rely upon HTTP redirects.
- 1002 • The interactions between the user agent and the identity provider are SOAP-based.

- The Liberty-enabled client and proxy profile includes Liberty-specified HTTP headers in the protocol messages it sends, signifying to identity providers and service providers that it is Liberty-enabled and thus can support capabilities beyond those supported by common non-Liberty-enabled user agents.

### 5.4.3.5 Single Sign-On Protocol Flow Example: Liberty Browser Artifact Profile

The first step in the single sign-on process in a Liberty browser artifact profile is that the user goes to a service provider and chooses to log in via the user's preferred identity provider. This login is accomplished by selecting the preferred identity provider from a list presented on the service provider's login page.

TECHNICAL NOTE: The service provider may discover the preferred identity provider via the identity provider introduction mechanism discussed 5.5 or, in the case of a Liberty-enabled client or proxy, by some other implementation-specific and unspecified means.

Once the user selects the identity provider, the user's browser is redirected to the identity provider with an embedded parameter indicating the originating service provider. The user can then log in to the identity provider as the user normally would. See Figure 20.

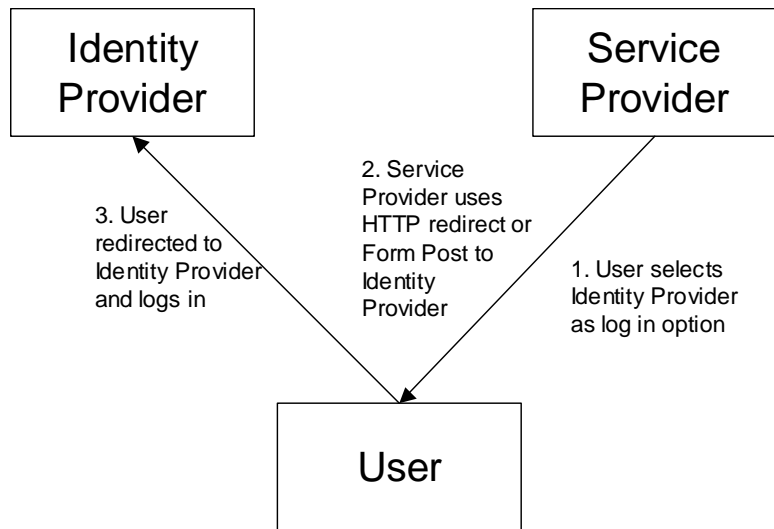
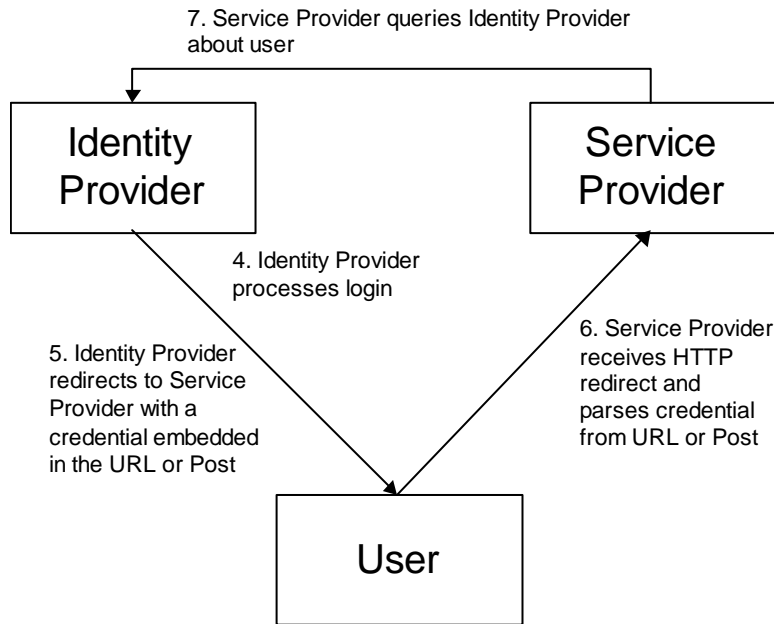


Figure 20: Single sign-on using HTTP redirect / form POST (1 of 2)

The identity provider then processes the login as normal and, upon successful login, redirects the user's browser back the originating service provider with a transient, encrypted credential, called an *artifact*, embedded within the URI. The service provider then parses the artifact from the URI and directly uses it to query the identity provider about the user. In its response, the identity provider vouches for the user, and the service provider may then establish a local notion of session state. See Figure 21.



1030  
1031

Figure 21: Single sign-on using HTTP redirect / form POST (2 of 2)

## 1032 5.5 Identity Provider Introduction

1033 In circle of trusts having more than one identity provider, service providers need a means to discover  
1034 which identity providers a user is using. Ideally, an identity provider could write a cookie that a  
1035 service provider could read. However, due to the cookie constraint outlined in 5.1.3, an identity  
1036 provider in one DNS domain has no standardized way to write a cookie that a service provider in  
1037 another DNS domain can read.

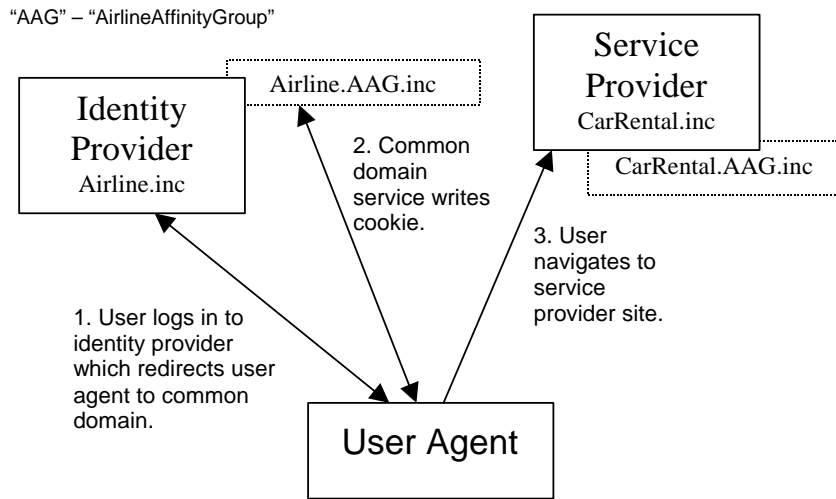
1038

1039 A solution to this introduction problem is to use a domain common to the circle of trust in question  
1040 and thus accessible to all parties, for example, AirlineAffinityGroup.inc or AAG.inc. Entries within  
1041 this DNS domain will point to IP addresses specified by each affinity group member. For example,  
1042 service provider CarRental.inc might receive a third-level domain “CarRental.AAG.inc” pointing to  
1043 an IP address specified by CarRental.inc. The machines hosting this *common domain service* would  
1044 be stateless. They would simply read and write cookies based on parameters passed within redirect  
1045 URLs.

1046

1047 When a user authenticates with an identity provider, the identity provider would redirect the user’s  
1048 browser to the identity provider’s instance of a common domain service with a parameter indicating  
1049 that the user is using that identity provider. The common domain service writes a cookie with that  
1050 preference and redirects the user’s browser back to the identity provider. Then, the user can navigate  
1051 to a service provider within the circle of trust. See Figure 22.

1052



1053

1054

Figure 22: Using a common domain to facilitate introductions (1 of 2)

1055

1056

1057

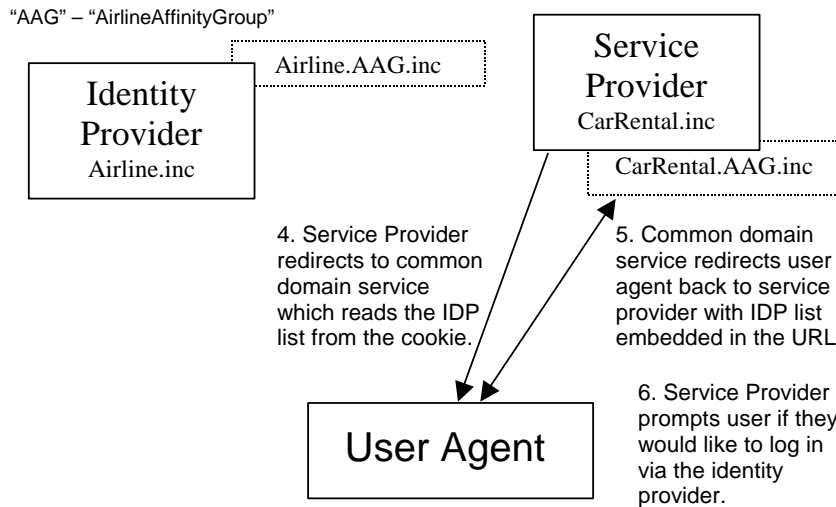
1058

1059

1060

1061

When the user navigates to a service provider within the circle of trust, the service provider can redirect the user's browser to its instance of the common domain service, which reads the cookie and redirects the user's browser back to the service provider with the user's identity provider embedded in the URL and thus available to service provider systems operating within the service provider's typical DNS domain. See Figure 23.



1062

1063

Figure 23: Using a common domain to facilitate introductions (2 of 2)

1064

1065

1066

1067

1068

The service provider now knows with which identity provider the user has authenticated within its circle of trust and can engage in further Liberty protocol operations with that identity provider, for example, single sign-on, on the user's behalf.

1069 POLICY/SECURITY NOTE:

1070

1071 **Common Domain Cookie Implications**

1072

1073 The identity provider can create either a session common domain cookie (for example, *this session only*; in  
1074 practice having ephemeral behavior, see [RFC2965]) or a persistent common domain cookie. The implications  
1075 with a session cookie are that it will disappear from the user agent cookie cache when the user logs out  
1076 (although this action would have to be explicitly implemented) or when the user agent is exited. This feature  
1077 may inconvenience some users. However, whether to use a session or a persistent cookie could be materialized  
1078 to the user at identity provider login time in the form of a Remember Me checkbox. If not checked, a session  
1079 cookie is used; if checked, a persistent one is used.

1080

1081 A user security implication of the persistent cookie is that if another person uses the machine, even if the user  
1082 agent had been exited, the persistent common domain cookie is still present—indeed all persistent cookies are  
1083 present. See the policy/security note in 5.1.3.

1084

1085 However, if the only information contained in a common domain cookie is a list of identity providers—that is, it  
1086 does not contain any personally identifiable information or authentication information, then the resultant  
1087 security risk to the user from inadvertent disclosure is low.

1088

1089 **Common Domain Cookie Processing**

1090

1091 The manner in which the common domain cookie writing service manipulates the common domain cookie is  
1092 specified in 3.6.2 of [LibertyBindProf]. The identity provider with which the user most recently authenticated  
1093 should be the last one in the list of identity providers in the cookie. However, the manner in which service  
1094 providers interpret the common domain cookie and display choices to the user is unspecified. This lack of  
1095 specificity implies that service providers may approach it in various ways. One way is to display identity  
1096 providers in a list ordered in reverse to the order in the common domain cookie. This approach will nominally  
1097 be in order of most-recently used if the common domain cookie writing service is adhering to the above  
1098 guideline. Or, the service provider may display only the last identity provider in the list. Or the service provider  
1099 may display the identity providers in some other order, if needed for some reason(s).

1100 **5.6 Single Logout**

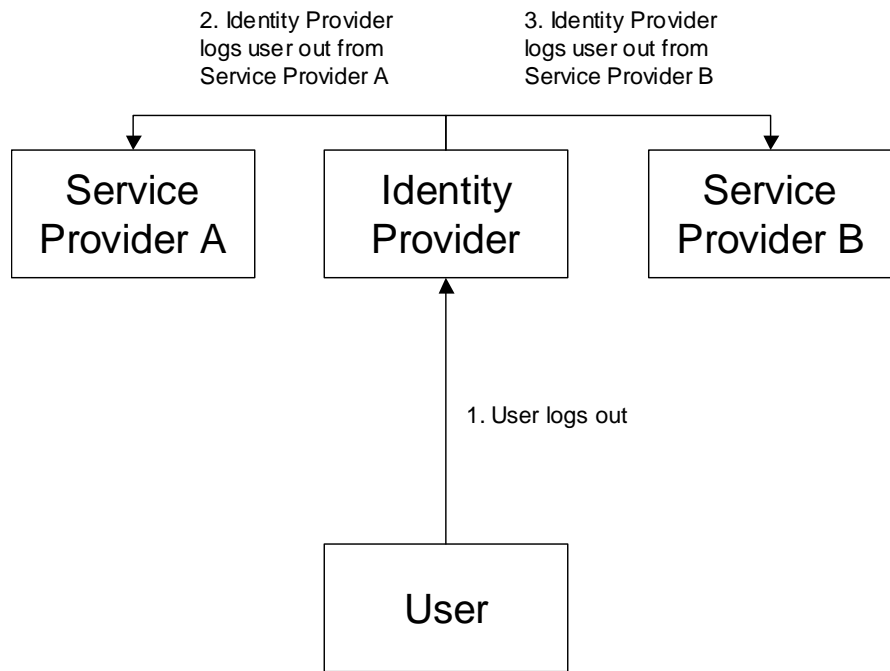
1101 The Single Logout Protocol and related profiles synchronize session logout functionality across all  
1102 sessions that were authenticated by a particular identity provider. The single logout can be initiated at  
1103 either the identity provider (see Figure 24) or the service provider (see Figure 25). In either case, the  
1104 identity provider will then communicate a logout notification to each service provider with which it  
1105 has established a session for the user.

1106

1107 POLICY/SECURITY NOTE: When using a single sign-on system, it is critical that, when users log out at a  
1108 service provider, their expectations are set about whether they are logging out from the identity provider or only  
1109 that particular service provider. It may be necessary to provide both Single Logout and Site Logout buttons or  
1110 links in Websites so that users' expectations are set. However, site logout may be regarded to come into play  
1111 only where users have to take a positive action to use their current authentication assertion at a site that they  
1112 have previously associated with their single sign-on.

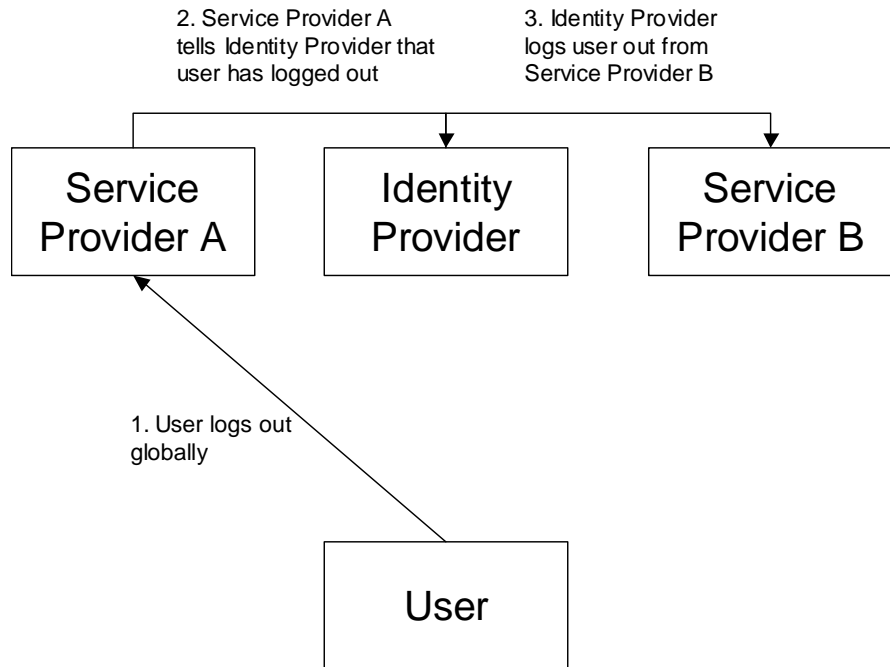
1113





1114  
1115  
1116

Figure 24: Single logout from an identity provider



1117  
1118  
1119

Figure 25: Single logout from a service provider

### 1120 5.6.1 Single Logout Profiles

1121 [LibertyBindProf] specifies three overall profiles for communicating the logout notification among  
1122 service providers and an identity provider:

1123

1124

- **HTTP-Redirect-Based:** Relies on using HTTP 302 redirects
- **HTTP-GET-Based:** Relies on using HTTP GET requests of IMG tags
- **SOAP/HTTP-Based:** Relies on asynchronous SOAP over HTTP messaging

1126

1127

All three profiles may be initiated at an identity provider. Only the first and the last may be initiated at a service provider. See [LibertyBindProf] for details.

1130

1131

TECHNICAL NOTE: The user-perceivable salient difference between the single logout profiles is that with the HTTP-redirect-based and SOAP/HTTP-based profiles, the Webpage from which the user initiates the logout process will remain in place as the logout process occurs (that is, each service provider is contacted in turn), while with the HTTP-GET-based profile, the identity provider has the opportunity to reload images (one per service provider, for example, completion check marks) on the viewed Webpage as the logout process proceeds.

1132

1133

1134

1135

## 1136 5.7 Example User Experience Scenarios

1137

This section presents several example user experience scenarios based upon the federation, introduction, and single sign-on facets of the Liberty Version 1.0 architecture. The intent is to illustrate the more subtle aspects of the user experience at login time and to illustrate commonWeb-specific user interface techniques that may be employed in prompting for, and collecting, the user's credentials. Specific policy and security considerations are called out.

1138

1139

1140

1141

### 1142 5.7.1 Scenario: Not Logged in Anywhere, No Common Domain Cookie

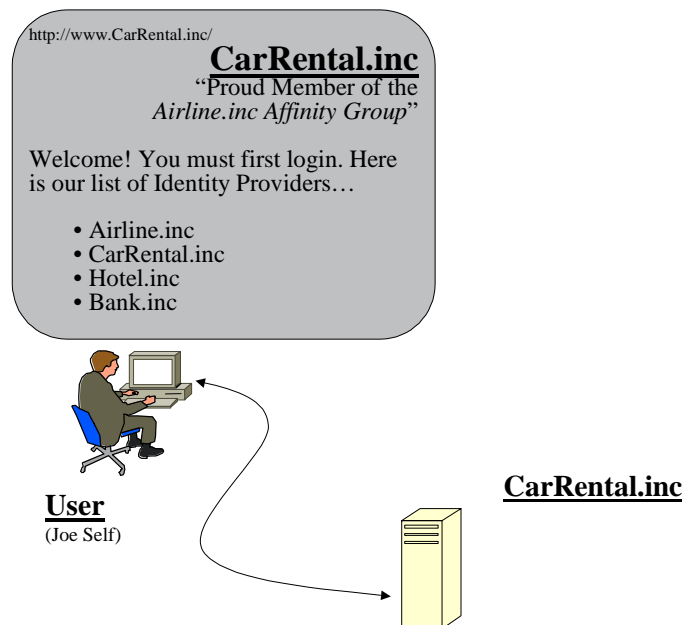
1143

In this scenario, Joe Self is not logged in at any Website, does not have a common domain cookie (for example, he restarted his user agent and/or flushed the cookie cache), and surfs to CarRental.inc. without first visiting his identity provider, Airline.inc.

1144

1145

1146



1147

1148

1149 **Figure 26: User arrives at service provider's Website without any authentication evidence or common domain cookie**

1149

1150

1151 CarRental.inc presents Joe Self with a welcome page listing identity providers from which he can  
1152 select (see Figure 26). Joe Self selects Airline.inc from the list.

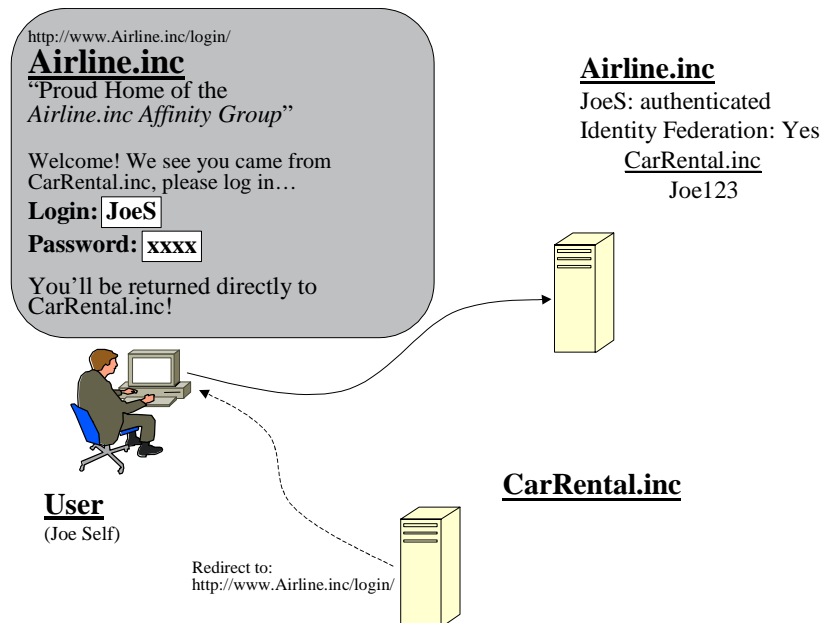
1153  
1154 Sections 5.7.1.1 through 5.7.1.3 illustrate three different, plausible, Web-specific user interface  
1155 techniques CarRental.inc, working in concert with Airline.inc, may use to facilitate Joe Self's login:  
1156

- 1157 • Redirect to identity provider Website
- 1158 • Identity provider dialog box
- 1159 • Embedded form

1160  
1161 TECHNICAL NOTE: These user interface techniques are commonly employed in Web-based systems. They are  
1162 not particular to, or specified by, Liberty. They are presented for illustrative purposes only.

### 1163 5.7.1.1 Login via Redirect to Identity Provider Website

1164 With login via redirect to the identity provider's Website, service providers provide direct links,  
1165 likely effected via redirects, to the identity provider's appropriate login page. Joe Self's browser will  
1166 display an identity provider's Webpage (see Figure 27); and upon successful login, his browser will  
1167 be redirected back to the service provider's Website where Joe Self will be provided access (see  
1168 Figure 30).  
1169



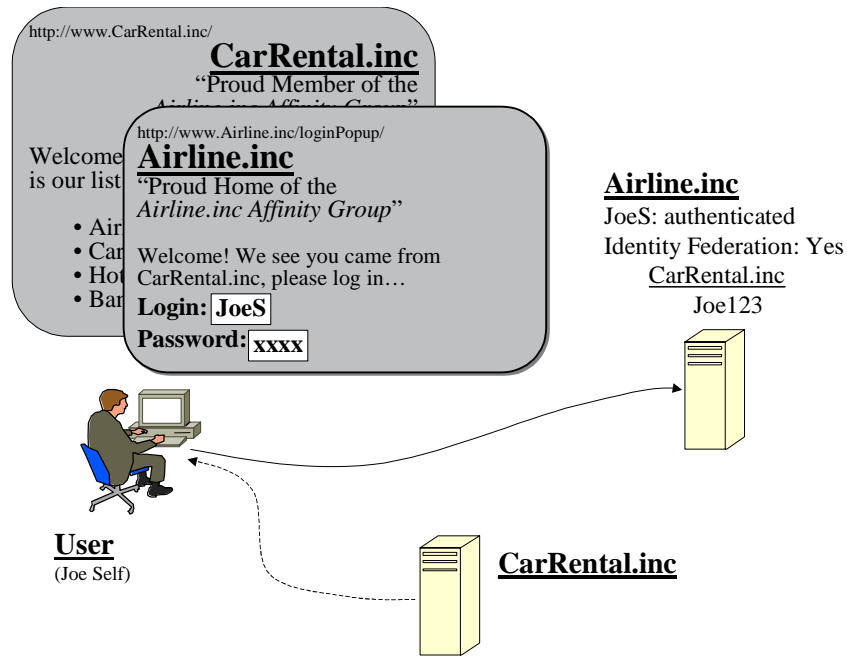
1170  
1171 **Figure 27: Service provider redirects to identity provider's login page.**

1172  
1173 POLICY/SECURITY NOTE: Login via redirect to the identity provider's Website is relatively secure in that the  
1174 user reveals his credentials directly to the identity provider. Of course, the usual security considerations  
1175 surrounding login and authentication events apply.

### 1176 5.7.1.2 Login via Identity Provider Dialog Box

1177 With login via a dialog box from the identity provider, the links on the service provider's Webpage  
1178 invoke a dialog or popup box. Joe Self's browser will display an identity provider popup (see Figure

1179 28); and upon successful login, the popup box will close, and Joe Self will be provided access at the  
1180 service provider's Website (see Figure 30).  
1181

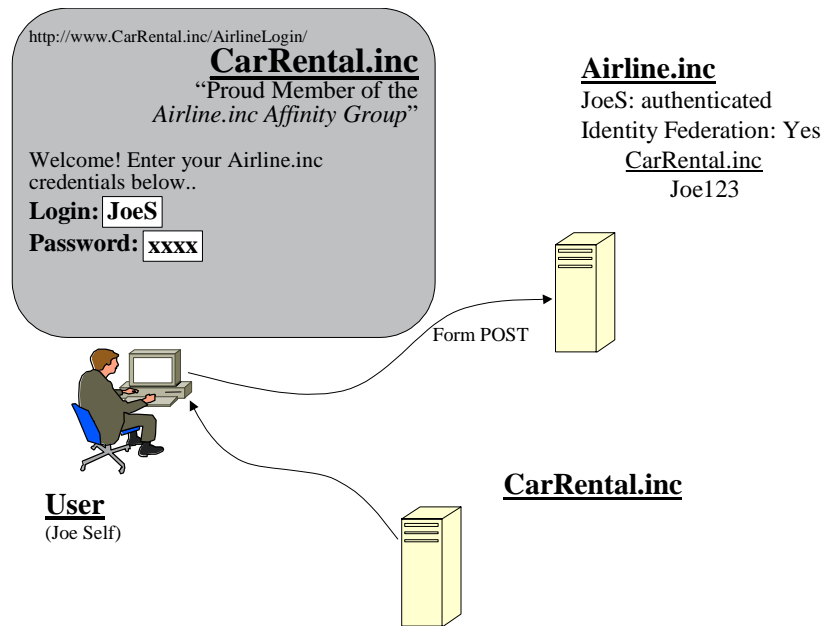


1182  
1183 **Figure 28: Service provider invokes dialog or popup box from identity provider.**

1184  
1185 POLICY/SECURITY NOTE: Login via a dialog box from the identity provider is relatively secure in that the  
1186 user reveals his credentials directly to the identity provider. Of course, the usual security considerations  
1187 surrounding login and authentication events apply.

### 1188 5.7.1.3 Login via Embedded Form

1189 With login via embedded form, the links on the service provider's Webpage cause the service  
1190 provider to display embedded login forms. In other words, the displayed page comes from the  
1191 service provider, but when Joe Self presses the Submit button, the information is conveyed to the  
1192 identity provider, typically via POST (see Figure 29). To Joe Self, it appears as if he has not left the  
1193 service provider's Webpages. Upon successful login, Joe Self will be provided access at the service  
1194 provider's Website (see Figure 30).  
1195



1196

1197

Figure 29: Login via embedded form

1198

1199

1200

1201

1202

1203

1204

1205

POLICY/SECURITY NOTE: Although users may like the seamlessness of this embedded form mechanism and deployers will like that the user does not leave their Website, it has serious policy and security considerations. In this mechanism, the user is revealing his identity provider credentials to the service provider in cleartext. Thus privacy surrounding the user's identity provider account is compromised. Additionally, a rogue service provider can now wield those credentials and impersonate the user. Thus, when using authentication via embedded form, deployers may want to consider appropriate contract terms between identity providers and service providers to address this risk.

1206

#### 5.7.1.4 The User is Logged in at CarRental.inc

1207

1208

1209

1210

CarRental.inc and Airline.inc then work in conjunction to effect login, and the CarRental.inc Website establishes a session based upon Joe Self's identity federation with Airline.inc (see Figure 30).

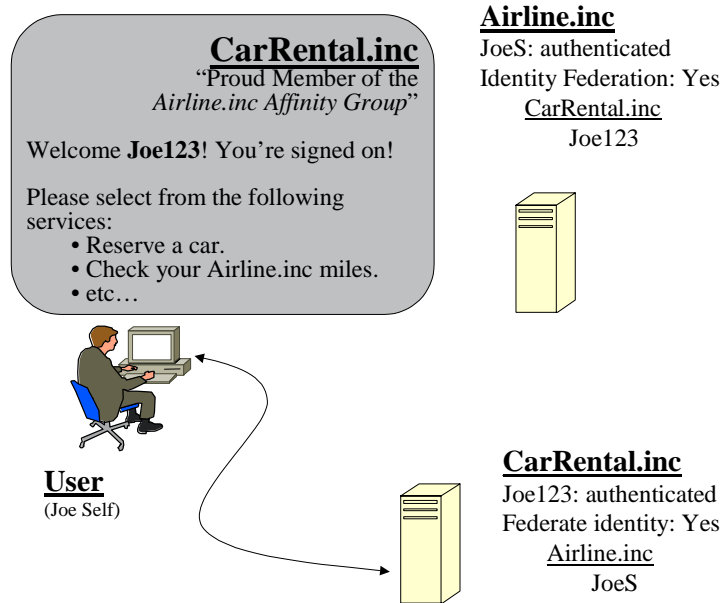


Figure 30: Service provider's Website delivers services on basis of federated identity.

1211  
1212  
1213

### 5.7.2 Scenario: Not Logged in Anywhere, Has a Common Domain Cookie

This scenario is similar the prior one. The only difference is that Joe Self's browser already has a common domain cookie cached. Therefore, when he arrives at a CarRental.inc Webpage, CarRental.inc will immediately know with which identity provider Joe Self is affiliated (Airline.inc in this case). It can immediately perform login via one of the three mechanisms outlined in the prior example or may prompt the user first.

1220  
1221  
1222  
1223  
1224

POLICY/SECURITY NOTE: Implementors and deployers should make allowance for the user to decide whether to immediately authenticate with the identity provider or be offered the chance to decline and authenticate either locally with the service provider or select from the service provider's list of affiliated identity providers.

### 5.7.3 Scenario: Logged in, Has a Common Domain Cookie

This scenario is the one illustrated in 2.2.

## 6 References

- 1228 [LibertyArchImpl] Kannappan, L., "Liberty Architecture Implementation Guidelines."
- 1229 [LibertyAuthnContext] Madsen, P., "Liberty Authentication Context Specification."
- 1230 [LibertyBindProf] Rouault, J., "Liberty Bindings and Profiles Specification."
- 1231 [LibertyGloss] Mauldin, H., "Liberty Glossary."
- 1232 [LibertyProtSchema] Beatty, J., "Liberty Protocols and Schemas Specification."
- 1233 [RFC1738] "Uniform Resource Locators (URL)," <http://www.ietf.org/rfc/rfc1738.txt>

- 1234 [RFC2119] S. Bradner, “Key words for use in RFCs to Indicate Requirement Levels,”  
1235 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 1236 [RFC2246] “The TLS Protocol Version 1.0,” <http://www.ietf.org/rfc/rfc2246.html>.
- 1237 [RFC2396] “Uniform Resource Identifiers (URI): Generic Syntax,”  
1238 <http://www.ietf.org/rfc/rfc2396.txt>.
- 1239 [RFC2616] “Hypertext Transfer Protocol — HTTP/1.1,” <http://www.ietf.org/rfc/rfc2616.txt>.
- 1240 [RFC2617] “HTTP Authentication,” <http://www.ietf.org/rfc/rfc2617.txt>.
- 1241 [RFC2965] “HTTP State Management Mechanism,” <http://www.ietf.org/rfc/rfc2965.txt>.
- 1242 [SAMLBind] P. Mishra et al., “Bindings and Profiles for the OASIS Security Assertion Markup  
1243 Language (SAML),” [http://www.oasis-open.org/committees/security/docs/draft-  
1244 sstc-bindings-model-11.pdf](http://www.oasis-open.org/committees/security/docs/draft-sstc-bindings-model-11.pdf), OASIS, January 2002.
- 1245 [SOAP1.1] D. Box et al., “Simple Object Access Protocol (SOAP) 1.1,”  
1246 <http://www.w3.org/TR/SOAP>, World Wide Web Consortium Note, May 2000.
- 1247 [SSLv3] “The SSL Protocol Version 3.0,”  
1248 <http://www.mozilla.org/projects/security/pki/nss/ssl/draft302.txt>.
- 1249