



Liberty ID-FF 1.2 Static Conformance Requirements

Version: 1.0

Editors:

Eric Tiffany, IEEE-ISTO

Contributors:

John Kemp, IEEE-ISTO

Abstract:

Defines the static conformance requirements for the Liberty Alliance ID-FF version 1.2 specifications.

Filename: liberty-idff-1.2-scr-v1.0.pdf

1 **Notice**

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

6 Implementation of certain elements of this document may require licenses under third party intellectual property
7 rights, including without limitation, patent rights. The Sponsors of and any other contributors to the Specification are
8 not, and shall not be held responsible in any manner for identifying or failing to identify any or all such third party
9 intellectual property rights. **This Specification is provided "AS IS", and no participant in the Liberty Alliance**
10 **makes any warranty of any kind, express or implied, including any implied warranties of merchantability,**
11 **non-infringement of third party intellectual property rights, and fitness for a particular purpose.** Implementors
12 of this Specification are advised to review the Liberty Alliance Project's website (<http://www.projectliberty.org>) for
13 information concerning any Necessary Claims Disclosure Notices that have been received by the Liberty Alliance
14 Management Board.

15 Copyright © 2004 ActivCard; America Online, Inc.; American Express Travel Related Services; Axalto; Bank of
16 America Corporation; Bell Canada; Cingular Wireless; Cisco Systems, Inc.; Communicator, Inc.; Deloitte & Touche
17 LLP; Earthlink, Inc.; Electronic Data Systems, Inc.; Entrust, Inc.; Epok, Inc.; Ericsson; Fidelity Investments; France
18 Telecom; Gemplus; General Motors; Hewlett-Packard Company; i2 Technologies, Inc.; Internet2; Intuit Inc.;
19 MasterCard International; NEC Corporation; Netegrity, Inc.; NeuStar, Inc.; Nextel Communications; Nippon
20 Telegraph and Telephone Corporation; Nokia Corporation; Novell, Inc.; NTT DoCoMo, Inc.; OneName Corporation;
21 Openwave Systems Inc.; Phaos Technology; Ping Identity Corporation; PricewaterhouseCoopers LLP; RegistryPro,
22 Inc.; RSA Security Inc; Sabre Holdings Corporation; SAP AG; SchlumbergerSema; Sigaba; SK Telecom; Sony
23 Corporation; Sun Microsystems, Inc.; Symlabs, Inc.; Trustgenix; United Airlines; VeriSign, Inc.; Visa International;
24 Vodafone Group Plc; Wave Systems. All rights reserved.

25 Liberty Alliance Project
26 Licensing Administrator
27 c/o IEEE-ISTO
28 445 Hoes Lane
29 Piscataway, NJ 08855-1331, USA
30 info@projectliberty.org

31 **Contents**

32 [1. Overview](#) 4
33 [2. Conformance Profiles](#) 5
34 [References](#) 31

35 1. Overview

36 Static conformance requirements (SCR) describe features that are mandatory and optional for implementations
37 conforming to the Liberty Alliance Identity Federation Framework Specifications (ID-FF version 1.2). This document
38 defines these requirements. This is a normative document with several non-normative explanatory sections.

39 1.1. Definitions and Motivation

40 The Liberty specifications define a large number of features and variations. Applications often do not require all
41 the features within a specification. It is also possible that implementations may not be able to implement all the
42 features. In these cases, it may be desirable to partition the specifications into subsets of functionality. A profile is
43 a subset of the overall specifications that includes all of the functionality necessary to satisfy the requirements of a
44 particular community of users. This document identifies several Liberty conformance profiles based on subsets of
45 these specifications according to the following guidelines:

- 46 • The number of conformance profiles should be kept small.
- 47 • The profiles should correspond to the major roles within the specifications: Identity Provider (IdP), Service
48 Provider (SP), and Liberty Enabled Client/Proxy (LECP).
- 49 • The SCR should distinguish between software implementations and deployments. Allow deployments to option-
50 ally configure features that are mandatory in the conformance profiles.
- 51 • The SCR should place more stringent requirements on IDPs as compared with SPs to promote interoperability.
- 52 • Implementations conforming to one (or more) profiles should be able to interoperate with another conforming
53 implementation of a complementary profile.

54 The resulting profiles are described below by means of tables indicating mandatory and optional features, with
55 references to the appropriate sections of the specification documents.

56 1.2. Notation

57 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD
58 NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in
59 [RFC2119]: "they MUST only be used where it is actually required for interoperation or to limit behavior which has
60 potential for causing harm (e.g., limiting retransmissions)."

61 The normative sections of this document are identified with special formatting as indicated here:

62 **EXAMPLE** This is an example of the format of the normative requirements. Each requirement is identified
63 by a requirement identifier ("EXAMPLE" in this example).

64 2. Conformance Profiles

65 2.1. Profile Matrices

66 The following tables summarize the features that comprise the conformance profiles. The detailed specifications for
 67 each profile follow in subsequent sections. The first table describes the four profiles derived from the original IDFF
 68 1.1 SCR document. The second table describes the profiles containing the IDFF 1.2 extensions.

69 **Table 1. Profile Matrix**

Feature	IDP	SP Basic	SP	LECP
Single Sign-On using Artifact Profile	MUST	MUST	MUST	
Single Sign-On using Browser POST Profile	MUST	MUST	MUST	
Single Sign-On using LECP Profile	MUST	MUST	MUST	MUST
Register Name Identifier - (IdP Initiated) - HTTP-Redirect	OPTIONAL	MUST	MUST	
Register Name Identifier - (IdP Initiated) - SOAP/HTTP	OPTIONAL	OPTIONAL	MUST	
Register Name Identifier - (SP Initiated) - HTTP-Redirect	MUST	MUST	MUST	
Register Name Identifier - (SP Initiated) - SOAP/HTTP	MUST	OPTIONAL	MUST	
Federation Termination Notification (IdP Initiated) – HTTP-Redirect	MUST	MUST	MUST	
Federation Termination Notification (IdP Initiated) – SOAP/HTTP	MUST	OPTIONAL	MUST	
Federation Termination Notification (SP Initiated) – HTTP-Redirect	MUST	MUST	MUST	
Federation Termination Notification (SP Initiated) – SOAP/HTTP	MUST	OPTIONAL	MUST	
Single Logout (IdP Initiated) – HTTP-Redirect	MUST	MUST	MUST	
Single Logout (IdP Initiated) – HTTP-GET	MUST	MUST	MUST	
Single Logout (IdP Initiated) – SOAP	MUST	OPTIONAL	MUST	
Single Logout (SP Initiated) – HTTP-Redirect	MUST	MUST	MUST	
Single Logout (SP Initiated) – SOAP	MUST	OPTIONAL	MUST	
Identity Provider Introduction (cookie)	MUST	OPTIONAL	OPTIONAL	
Backward Compatibility	OPTIONAL	OPTIONAL	OPTIONAL	

70 The following table summarizes additional profiles which extend the profiles defined above. These extended profiles
 71 are to be understood as a combination of an IDP or SP profile from the table above with a corresponding extended
 72 profile below.

73 **Table 2. Extended Profile Matrix**

Feature	IDP Extended	SP Extended
One-Time (Anonymous) Name Identifiers	MUST	MUST
Affiliations	MUST	MUST
Identity Provider Proxy	MUST	MUST
Name Identifier Mapping	MUST	MUST

74 2.2. Identity Provider Conformance Profile

75 This section defines the conformance requirements for an identity provider. These requirements are derived from the
 76 steps defined in the [[LibertyBindProf, Section 3](#)], which describe the interactions between the user agent, the service
 77 provider, and the identity provider.

78 2.2.1. Single Sign-On and Federation

79 This section defines how an identity provider facilitates single sign-on by processing incoming and outgoing requests.
80 The steps indicated refer to the interaction diagram in [LibertyBindProf, 3.2.1] which illustrates the general single
81 sign-on framework.

82 **2.2.1.1. Common Interaction and Processing**

83 The single sign-on requirements specified here assume that the user agent has already authenticated with the identity
84 provider, and that a valid session exists for the user agent at the identity provider.

85 The following actions are required of the identity provider:

86 **IDP-SSO-001** In step 5, the identity provider **MUST** process the `<lib:AuthnRequest>` message according to
87 the rules specified in [LibertyProtSchema].

88 **IDP-SSO-002** In step 6, the identity provider **MUST** respond to the user agent with a
89 `<lib:AuthnResponse>`, a SAML artifact, or an error. The form of this response is
90 contingent on the specific interaction method employed by the identity provider.

91 **IDP-SSO-014** The identity provider **MUST** return an appropriately scoped `<NameIdentifier>` in accordance
92 with the rules specified for the `<NameQualifier>` in [LibertyProtSchema section
93 3.2.2.3].

94 **IDP-SSO-015** The identity provider **MUST** return a `<NameIdentifier>` corresponding to the
95 `<NameIDPolicy>` specified in the authentication request [LibertyProtSchema section
96 3.2.2.6].

97 **IDP-SSO-016** If the identity provider wishes to proxy authentication to another authentication provider, they
98 **MUST** follow the rules specified in [LibertyProtSchema section 3.2.2.6]. to generate the
99 proxied authentication request.

100 **2.2.1.2. Single Sign-on Using Browser Artifact**

101 This section describes the identity provider actions necessary to perform single sign-on using browser artifact. The
102 requirements in this section are derived primarily from Liberty Bindings and Profiles, Section 3.2.2, with additional
103 references to Step 8 in Section 3.1.

104 **IDP-SSO-003** Single Sign-on using Browser Artifact is a mandatory supported feature of the identity provider
105 conformance profile. The requirements in this section **MUST** be implemented according to the
106 relevant sections of [LibertyBindProf].

107 The identity provider must complete two processing steps to implement this feature: processing an authentication
108 request, and processing a SAML assertion. The authentication interaction proceeds as follows:

109 **IDP-SSO-004** The identity provider **MUST** process the `<lib:AuthnRequest>` message as specified in
110 [LibertyBindProf, 3.2.1.5, Step 5].

111 **IDP-SSO-005** In response to the `<lib:AuthnRequest>` the identity provider **MUST** perform a redirection
112 as specified in [LibertyBindProf, 3.2.2.1.2, Step 6]

113 **IDP-SSO-006** The identity provider **MUST** process the `<samlp:Request>` produced by the service provider
114 in Step 8 of the single sign-on interaction, and produce a `<samlp:Response>` as specified in
115 [\[LibertyBindProf, 3.2.1.9, Step 9\]](#).

116 **IDP-SSO-007** The artifact produce by the identity provider **MUST** be be formatted as specified in [\[Liberty-](#)
117 [BindProf, 3.2.2.2, Artifact Format\]](#).

118 **2.2.1.3. Single Sign-on using Liberty Browser POST**

119 This section describes the identity provider requirements for performing single sign-on using Liberty Browser POST.

120 **IDP-SSO-008** Single Sign-on using Liberty Browser POST is a mandatory supported feature of the identity
121 provider conformance profile. The requirements in this section **MUST** be implemented
122 according the the relevant sections of [\[LibertyBindProf\]](#).

123 **IDP-SSO-009** The identity provider **MUST** process the `<lib:AuthnRequest>` message as specified in
124 [\[LibertyBindProf, 3.2.1.5, Step 5\]](#). (Same as [\[IDP-SSO-004\]](#)).

125 **IDP-SSO-010** The identity provider generates an HTML 200 OK response containing an authentication
126 request. This response **MUST** conform to the specification in [\[LibertyBindProf, 3.2.3.2, Step](#)
127 [6\]](#)

128 **2.2.1.4. Single Sign-on using Liberty-Enabled Client and Proxy**

129 This section specifies the identity provider requirements for performing single sign-on using the Liberty-Enabled
130 Client and Proxy (LECP) interaction.

131 **IDP-SSO-011** Single Sign-on using LECP is a mandatory supported feature of the identity provider confor-
132 mance profile. The requirements in this section **MUST** be implemented according the the
133 relevant sections of [\[LibertyBindProf, 3.2.4\]](#).

134 **IDP-SSO-012** The identity provider **MUST** process the `<lib:AuthnRequest>` in the body of the SOAP
135 POST message from the LECP as specified in [\[LibertyBindProf, 3.2.1.5, Step 5\]](#). (See [\[IDP-](#)
136 [SSO-004\]](#)).

137 **IDP-SSO-013** The identity provider **MUST** respond to the `<lib:AuthnRequest>` with a HTTP 200
138 OK response as specified in [\[LibertyBindProf, 3.2.4.2.4, Step 6\]](#), with the correct MIME
139 type (`application/vnd.liberty-response+xml`) and Liberty-Enabled HTTP header (see
140 [\[LibertyBindProf, 3.2.4.1\]](#))

141 **2.2.2. Register Name Identifier**

142 This section specifies the required and optional features used by an identity provider to register or change a name
143 identifier for a Principal. There are four variations of the register name identifier protocol: the name registration
144 interaction can be initiated by either the identity provider or the service provider, and the protocol can be either HTTP-
145 Redirect based, or SOAP/HTTP based [[LibertyBindProf, 3.3](#)].

146 Note that while the name registration features are optional interactions in a *deployment*, some of these features are
147 required for a implementation to be conformant. The following sections describe the mandatory and optional
148 conformance requirements for an identity provider implementing the register name identifier feature.

149 **2.2.2.1. Register Name Identifier Initiated at Identity Provider**

150 **2.2.2.1.1. HTTP-Redirect Based**

151 This section specifies the requirements for the HTTP-Redirect-based register name identifier initiated at the identity
152 provider. These requirements are based on the identity provider actions in the interaction described in [[LibertyBind-
153 Prof, 3.3.1.1](#)]. Note that the timing and mechanism of the initiation of this interaction are not normatively specified,
154 although the preceding specification reference offers some examples.

155 **IDP-RNI-001** The identity provider **MUST** redirect the user agent to the register name identifier service at the
156 service provider as specified in [[LibertyBindProf, 3.3.1.1.2, Step 2](#)].

157 **2.2.2.1.2. SOAP/HTTP Based**

158 **IDP-RNI-002** The identity provider **MUST** only initiate SOAP/HTTP-based register name identifier when the
159 service provider metadata specifies the appropriate URI identifier as specified in [[LibertyBind-
160 Prof, 3.3.1.2](#)].

161 **IDP-RNI-003** The SOAP/HTTP-based Register Name Identifier transactions must use the SOAP binding for
162 Liberty as defined in [[LibertyBindProf, 2.1](#)].

163 **IDP-RNI-004** The identity provider **MUST** initiate the Register Name Identifier transaction by sending a
164 `<lib:RegisterNameIdentifierRequest>` message to the service provider's SOAP end-
165 point as specified in [[LibertyBindProf, 3.3.1.2.1, Step 1](#)].

166 **IDP-RNI-005** The identity provider **MUST** process the `<lib:RegisterNameIdentifierResponse>` from
167 the service provider as specified in [[LibertyProtSchema, 3.3.3](#)].

168 **2.2.2.2. Register Name Identifier Initiated at Service Provider**

169 The following sections describe the interactions for an identity provider implementing the Register Name Identifier
170 initiated at the service provider.

171 **IDP-RNI-006** Register Name Identifier initiated at service provider is a **REQUIRED** feature of the Identity
172 Provider Conformance Profile. Both the HTTP-Redirect and the SOAP/HTTP interaction
173 **MUST** be implemented.

174 Note that this section refers to [\[LibertyBindProf\]](#) Register Name Identifier interactions initiated at the identity provider.
175 The steps associated with the service provider in those interactions are used here as if they were associated with the
176 identity provider. All references to service provider and identity provider have been interchanged as indicated in
177 [\[LibertyBindProf, 3.3.2.1\]](#) and [\[LibertyBindProf, 3.3.2.2\]](#).

178 **2.2.2.2.1. HTTP-Redirect Based**

179 **IDP-RNI-007** The identity provider MUST process the `<lib:RegisterNameIdentifierRequest>` from
180 the service provider as specified in [\[LibertyProtSchema, 3.3.3\]](#). See [\[LibertyBindProf,](#)
181 [3.3.1.1.4, Step 4\]](#).

182 **IDP-RNI-008** The identity provider MUST respond to the service provider with a redirection URL as specified
183 in the `RegisterNameIdentifierServiceReturnURL` metadata element. The redirection
184 MUST adhere to the rules specified in [\[LibertyBindProf, 3.3.1.1.5, Step 5\]](#).

185 **2.2.2.2.2. SOAP/HTTP Based**

186 [\[IDP-RNI-003\]](#) is a requirement for this interaction.

187 **IDP-RNI-009** The service provider will send a `<lib:RegisterNameIdentifierRequest>` proto-
188 col message to the identity provider. The identity provider MUST record the new
189 `<lib:SPProvidedNameIdentifier>`.

190 **IDP-RNI-010** After a successful registration of the `<lib:SPProvidedNameIdentifier>`, the identity
191 provider MUST respond with a `<lib:RegisterNameIdentifierResponse>` according to
192 the processing rules in [\[LibertyProtSchema, 3.3.3\]](#).

193 **2.2.3. Identity Federation Termination Notification**

194 Liberty identity federation termination notification specifies how service providers and identity providers are notified
195 of federation termination. There are four variations of the federation termination notification interaction: the
196 federation termination notification interaction can be initiated by either the identity provider or the service provider,
197 and the protocol can be based on either HTTP-Redirect or SOAP/HTTP.

198 **IDP-FTN-001** This section specifies the conformance requirements for an identity provider to support identity
199 federation termination notification. All four interactions specified in [\[LibertyBindProf, 3.4\]](#)
200 MUST be implemented.

201 **2.2.3.1. Federation Termination Notification Initiated at the Identity Provider**

202 **2.2.3.1.1. HTTP-Redirect**

203 **IDP-FTN-002** This interaction MUST NOT be used unless the service provider metadata ele-
204 ment `FederationTerminationNotificationProtocolProfile` specifies the URI
205 <http://projectliberty.org/profiles/fedterm-idp-http>.

206 **IDP-FTN-003** This interaction REQUIRES certain preconditions specified in [\[LibertyBindProf, 3.4.1.1\]](#) are
207 met.

208 **IDP-FTN-004** In response to a request to the identity provider’s federation termination service URL, the
209 identity provider **MUST** redirect the user agent to the federation termination service at the
210 service provider. This redirection **MUST** adhere to the rules specified in [[LibertyBindProf](#),
211 3.4.1.1.2, Step 2].

212 **2.2.3.1.2. SOAP/HTTP**

213 **IDP-FTN-005** This interaction **MUST NOT** be used unless the service provider metadata ele-
214 ment `FederationTerminationNotificationProtocolProfile` specifies the URI
215 <http://projectliberty.org/profiles/fedterm-idp-soap> .

216 **IDP-FTN-006** This interaction **REQUIRES** certain preconditions specified in [[LibertyBindProf](#), 3.4.1.2] are
217 met.

218 **IDP-FTN-007** In response to a request from the user agent to the identity provider’s federation termination
219 service URL, the identity provider **MUST** send an asynchronous SOAP over HTTP notification
220 message to the service provider’s SOAP endpoint. The SOAP message **MUST** adhere to the
221 rules specified in [[LibertyBindProf](#), 3.4.1.2.2, Step 2].

222 The service provider will respond to termination notification with a HTTP 204 No Content response.

223 **IDP-FTN-008** The identity provider **MUST** process the HTTP 204 No Content response from the service
224 provider and send an HTTP response confirming the requested action of federation termination
225 with the specified service provider.

226 **2.2.3.2. Federation Termination Initiated at the Service Provider**

227 The following sections describe the interactions for an identity provider implementing federation termination notifica-
228 tion initiated at the service provider.

229 Note that this section refers to [[LibertyBindProf](#)] federation termination notifications interactions initiated at the
230 identity provider. The steps associated with the service provider in those interactions are used here as if they
231 were associated with the identity provider. All references to to service provider and identity provider have been
232 interchanged as indicated in [[LibertyBindProf](#), 3.4.2.1] and [[LibertyBindProf](#), 3.4.2.2].

233 **2.2.3.2.1. HTTP-Redirect**

234 **IDP-FTN-009** The identity provider **MUST** process the `<lib:FederationTerminationNotification>`
235 received from the user agent according to the rules defined in [[LibertyProtSchema](#), 3.4.2] and
236 in [[LibertyBindProf](#), 3.4.1.1.4, Step 4].

237 **IDP-FTN-010** The identity provider’s federation termination service **MUST** respond by redirecting the user
238 agent as specified in [[LibertyBindProf](#), 3.4.1.1.5, Step 5].

239 **2.2.3.2.2. SOAP/HTTP**

240 **IDP-FTN-011** The identity provider **MUST** process the `<lib:FederationTerminationNotification>`
241 in the SOAP message received from the service provider according to the rules defined in
242 [[LibertyProtSchema](#), 3.4.2] and in [[LibertyBindProf](#), 3.4.1.2.3, Step 3].

243 **IDP-FTN-012** The identity provider **MUST** respond to the `<lib:FederationTerminationNotification>`
244 with a HTTP 204 OK response [[LibertyBindProf, 3.4.1.2.4, Step 4](#)].

245 **2.2.4. Single Logout**

246 Liberty single logout specifies how service providers and identity providers synchronize logout across all sessions
247 authenticated by a particular identity provider. There are five variations of the single logout interaction: the single
248 logout can be initiated by either the identity provider or the service provider, and the protocol can be based on either
249 HTTP-Redirect, HTTP-GET (only when initiated at the identity provider), or SOAP/HTTP.

250 **IDP-SLO-001** This section specifies the conformance requirements for an identity provider to support Single
251 Logout. All five interactions specified in [[LibertyBindProf, 3.5](#)] **MUST** be implemented.

252 **2.2.4.1. Single Logout Initiated at the Identity Provider**

253 The following sections specify the requirements for single logout when initiated by a user agent at the identity provider.

254 **2.2.4.1.1. HTTP-Redirect**

255 **IDP-SLO-002** This interaction **MUST NOT** be used unless the service provider metadata element
256 `SingleLogoutProtocolProfile` specifies the URI `http://projectliberty.org/profiles/slo-idp-`
257 `http`.

258 **IDP-SLO-003** In response to the user agent request, the identity provider **MUST** redirect the user agent to
259 the single logout service URL at each service provider for which the identity provider has
260 provided an authentication assertion during the Principal's current session. Each redirection
261 **MUST** adhere to the rules specified in [[LibertyBindProf, 3.5.1.1.2, Step 2](#)].

262 **IDP-SLO-004** After receiving the request from the user agent to the `SingleLogoutServiceReturnURL` as
263 specified in the identity provider metadata, the identity provider **MUST** process the request and
264 send an HTTP response to the user agent confirming that the requested action of a single logout
265 has been completed.

266 **2.2.4.1.2. HTTP-GET**

267 **IDP-SLO-005** This interaction **MUST NOT** be used unless the service provider metadata element
268 `SingleLogoutProtocolProfile` specifies the URI `http://projectliberty.org/profiles/slo-idp-`
269 `http`.

270 **IDP-SLO-006** In response to the user agent request, the identity provider **MUST** respond with a standard
271 HTTP 200 OK response containing image tags referencing the logout service URL for each of
272 the service providers for which the identity provider has provided an authentication assertion
273 during the Principal's current session. Each image tag **MUST** adhere to the rules specified in
274 [[LibertyBindProf, 3.5.1.1.2.2, Step 2](#)].

275 **IDP-SLO-007** After receiving the request from the user agent to the `SingleLogoutServiceReturnURL` as
276 specified in the identity provider metadata, the identity provider **MUST** process the request and
277 send an HTTP response to the user agent confirming that the requested action of a single logout
278 has been completed.

279 **2.2.4.1.3. SOAP/HTTP**

280 The requirements for the SOAP/HTTP single logout interaction are described in [[LibertyBindProf, 3.5.1.2](#)].

281 **IDP-SLO-008** This interaction **MUST NOT** be used unless the service provider metadata element
282 `SingleLogoutProtocolProfile` specifies the URI `http://projectliberty.org/profiles/slo-idp-`
283 `soap` .

284 **IDP-SLO-017** The identity provider **MUST** send a SOAP over HTTP request to the SOAP endpoint of each ser-
285 vice provider for which the identity provider provided authentication assertions during the Prin-
286 cipal's current session. Each message **MUST** contain exactly one `<lib:LogoutRequest>`
287 element in the SOAP body and adhere to the rules specified in [[LibertyBindProf, 3.5.1.2.2](#)] and
288 [[LibertyProtSchema, 3.5.1](#)].

289 **IDP-SLO-009** In response to a SOAP 200 OK `<lib:LogoutResponse>` message from the service provider,
290 the identity provider **MUST** send an HTTP response confirming the requested action of single
291 logout has completed.

292 **2.2.4.2. Single Logout Initiated at the Service Provider**

293 **2.2.4.2.1. HTTP-Redirect**

294 **IDP-SLO-010** The user agent will access the identity provider's single logout service URL. The identity
295 provider **MUST** process the `<lib:LogoutRequest>` according to the rules defined in [[Lib-](#)
296 [ertyProtSchema, 3.5.1](#)]

297 **IDP-SLO-011** The identity provider **MUST** notify each service provider for which the identity provider has
298 provided authentication assertions of the logout request via the service provider's preferred
299 profile [[LibertyBindProf, 3.5.2.1.4, Step 4](#)].

300 **IDP-SLO-012** The identity provider **MUST** terminate the Principal's current session, and no more authentica-
301 tion assertions for the Principal are to be given to service providers.

302 **IDP-SLO-013** The identity provider **MUST** respond and redirect the user agent back to the service provider us-
303 ing the return URL location obtained from the `SingleLogoutServiceReturnURL` metadata
304 element as specified in [[LibertyBindProf, 3.5.2.1.5, Step 5](#)].

305 **2.2.4.2.2. SOAP/HTTP**

306 **IDP-SLO-014** After receiving a `<lib:LogoutRequest>` from the service provider, the identity provider
307 **MUST** process it according to the rules in [[LibertyProtSchema, 3.5.1](#)].

308 **IDP-SLO-015** The identity provider **MUST** submit to each service provider for which the identity provider has
309 provided authentication assertions during the Principal's current session a request to logout the
310 Principal as specified in [[LibertyBindProf, 3.5.2.2.3, Step 3](#)].

311 **IDP-SLO-016** The identity provider **MUST** respond to the `<lib:LogoutRequest>` with a SOAP 200 OK
312 `<lib:LogoutResponse>` message [[LibertyBindProf, 3.5.2.2.4, Step 4](#)].

313 **2.2.5. Identity Provider Introduction**

314 This section describes the conformance requirements for an identity provider implementing the identity provider
315 introduction feature. The identity provider introduction feature is intended to allow service providers to discover
316 which identity providers a Principal is using.

317 **IDP-IPI-001** The identity provider introduction feature is a **REQUIRED** element of identity provider confor-
318 mance profile.

319 Although a deployment may choose not to enable the identity provider introduction feature, an identity provider
320 implementation must provide the feature in order to be conformant.

321 **2.2.5.1. Common Domain Cookie**

322 The identity provider introduction relies on the use of a common domain cookie.

323 **IDP-IPI-002** The common domain cookie **MUST** be constructed as specified in [[LibertyBindProf, 3.6.1](#)].

324 **2.2.5.2. Setting the Common Domain Cookie**

325 Creating and updating the common domain cookie is the responsibility of the identity provider.

326 **IDP-IPI-003** After authenticating a Principal, the identity provider **MUST** attempt to set the common domain
327 cookie, subject to cookie-setting restrictions of the user-agent.

328 The details of this procedure are implementation-dependent, and are not normatively specified. However, one possible
329 strategy is described in [[LibertyBindProf, 3.6.2](#)].

330 **2.2.6. Backward Compatibility with v1.1 Protocol**

331 Backward compatibility is optional for conformant identity providers:

332 **IDP-BCK-001** Backward compatibility with 1.1 service provider implementations is **OPTIONAL** for an
333 identity provider implementation. However, an implementation claiming backward compatible
334 conformance **MUST** adhere to the relevant rules as described in [[LibertyProtSchema, 3.1.11](#)].

335 **2.3. Extended Identity Provider Conformance Profile**

336 This section describes the requirements for an implementation to be an Extended Identity Provider. This profile
337 extends the Identity Provider Profile defined in the previous section.

338 **IDP-EXT-001** An Extended Identity Provider implementation **MUST** conform to the Identity Provider profile
339 requirements. In addition, an Extended Identity Provider **MUST** conform to all the require-
340 ments defined in the following subsections.

341 **2.3.1. One-Time (Anonymous) Name Identifier**

342 A service provider can request a one-time (also known as "anonymous") name identifier for a Principal during the
343 sign-on interaction with the identity provider by specifying a value of *onetime* for the `NameIDPolicy` element in the
344 `AuthnRequest`.

345 In this case, the identity provider will construct the `AuthnResponse` according to the following requirements:

346 **IDP-ONE-001** If the `<NameIDPolicy>` element is *onetime*, then the `<saml:NameIdentifier>` element in
347 the `<saml:Subject>` element **MUST** be a temporary, one-time-use identifier for the Principal,
348 with a `Format` attribute of *urn:liberty:iff:nameid:one-time*.

349 **IDP-ONE-002** The temporary identifier **MUST** be constructed according to the specifications in [[LibertyProtSchema](#), 3.1.4].
350

351 **2.3.2. Affiliations**

352 An affiliation is a set of one or more entities, described by `providerIDs`, who may perform Liberty interactions as a
353 member of the set. This section describes the conformance requirements for an identity provider implementing the
354 affiliation features of the IDFF 1.2 specifications. These requirements are derived from protocol processing rules in
355 [[LibertyProtSchema](#)].

356 **IDP-AFF-001** The value of the `<AffiliationID>` element of ID-FF 1.2 protocol messages **MUST** adhere to
357 the uniqueness constraints described in [[LibertyProtSchema](#), 3.1.3].

358 **2.3.2.1. Single Sign-on and <AuthnResponse>**

359 The identity provider will issue an `<AuthnResponse>` after processing a `<AuthnRequest>` from a service provider
360 (or, in some cases, an identity provider may issue an `<AuthnResponse>` without first receiving such a request). These
361 are the requirements that obtain when an affiliation is present in the request (see [[LibertyProtSchema](#), 3.2.2.6]).

362 **IDP-AFF-002** If the `<AffiliationID>` element is present, then the `<saml:NameIdentifier>` **MUST** be
363 the most recent name identifier provided by a member of the affiliation, if any, or the name
364 identifier for the Principal supplied by the identity provider for the affiliation.

365 **IDP-AFF-003** `<AffiliationID>`, if present, **MUST** be the unique identifier of a known affiliation group
366 with which the identity provider has an established relationship, and of which the requesting
367 provider is a member. If present, identity providers **MUST** establish and resolve federations
368 based on the specified affiliation, not the requesting provider. In addition, identity providers
369 **MAY** retrieve information regarding the other members of the affiliation group by querying
370 metadata (see [[LibertyMetadata](#)]) and present a list of members of the affiliation group to the
371 Principal.

372 **IDP-AFF-004** In all of the name identifier elements in the request and response messages of this protocol,
373 if the Principal's identity federation is between the identity provider and an affiliation group
374 in which the service provider is a member, then the `NameQualifier` attribute **MUST** contain
375 the unique identifier of the affiliation group. Otherwise, it **MUST** contain the unique identifier
376 of the service provider. This attribute **MUST** be used by the providers to identify the specific
377 identity federation to be modified.

378 **2.3.2.2. Name Registration**

379 The provider is required to correctly identify the affiliation in the <RegisterNameIdentifierRequest> and
380 <RegisterNameIdentifierResponse> messages.

381 **IDP-AFF-005** In all of the name identifier elements in the request and response messages of this protocol,
382 if the Principal's identity federation is between the identity provider and an affiliation group
383 in which the service provider is a member, then the `NameQualifier` attribute **MUST** contain
384 the unique identifier of the affiliation group. Otherwise, it **MUST** contain the unique identifier
385 of the service provider. This attribute **MUST** be used by the providers to identify the specific
386 identity federation to be modified.

387 See [[LibertyProtSchema, 3.3.3](#)].

388 **2.3.2.3. Federation Termination**

389 The provider is required to correctly identify the affiliation when performing federation termination using the
390 <FederationTerminationNotification> message.

391 **IDP-AFF-006** If the Principal's identity federation was between the identity provider and an affiliation group
392 in which the service provider is a member, then the `NameQualifier` attribute **MUST** contain
393 the unique identifier of the affiliation group. Otherwise, it **MUST** contain the unique identifier
394 of the service provider. This attribute **MUST** be used by the providers to identify the specific
395 identity federation being terminated.

396 See [[LibertyProtSchema, 3.4.2](#)].

397 **2.3.2.4. Single Logout**

398 The provider is required to correctly identify the affiliation when responding to a single logout request using the
399 <LogoutResponse> message.

400 **IDP-AFF-007** If the Principal's identity federation is between the identity provider and an affiliation group
401 in which the service provider is a member, then the `NameQualifier` attribute **MUST** contain
402 the unique identifier of the affiliation group. Otherwise, it **MUST** contain the unique identifier
403 of the service provider. This attribute **MUST** be used by the providers to identify the specific
404 identity federation of the Principal who is logging out.

405 See [[LibertyProtSchema, 3.5.2.3](#)].

406 **2.3.3. Dynamic Proxy of Identity Provider**

407 A service provider may request that an identity provider obtain authentication for a Principal from a third-party identity
408 provider. This section describes the conformance requirements for an identity provider implementing this dynamic
409 proxy feature.

410 **IDP-PXY-001** A conformant identity provider implementation **MUST** adhere to the processing rules defined
411 in [[LibertyProtSchema, 3.2.2.7](#)].

412 However, note that the dynamic proxy specifications do not compel an identity provider to perform a proxy interaction.
413 For purposes of conformance, the following directives are made compulsory. In an actual deployment, whether or not
414 to proxy would be a local policy issue.

- 415 **IDP-PXY-002** The identity provider **MUST** proxy an authentication request if the value in the <ProxyCount>
416 element is greater than zero, or if no <ProxyCount> appears in the request.
417 The identity provider **MUST** proxy for a provider specified in the <IDPList>.

418 **2.3.4. Name Identifier Mapping**

419 This section describes the conformance requirements for an identity provider performing NameIdentifier mapping.
420 The requirements in this section are derived from [[LibertyBindProf, 3.7](#)]. The SOAP-based NameIdentifier interaction
421 is the sole interaction described in the specification.

- 422 **IDP-NIM-001** The SOAP-based NameIdentifier Mapping interaction is a mandatory supported feature of the
423 Extended Identity Provider Conformance Profile. The requirements in this section **MUST** be
424 implemented according to the relevant section of [[LibertyBindProf](#)].

425 The NameIdentifier mapping interaction specifies how one service provider (the "requester") may obtain a NameIdentifier
426 for a Principal it has federated in the namespace of a different service provider (the "target"). The "requester"
427 accomplishes this by issuing a <lib:NameIdentifierMappingRequest> to an identity provider that has federated
428 the Principal with both service providers (see [[LibertyBindProf, Figure 17 and 3.7.1.1](#)])

- 429 **IDP-NIM-002** The identity provider **MUST** process the <lib:NameIdentifierMappingRequest> as indicated
430 in [[LibertyBindProf, 3.7.1.2, Step 2](#)] and in [[LibertyProtSchema, 3.6.2 and 3.6.3](#)].

431 Specifically:

- 432 **IDP-NIM-003** The identity provider **MUST** validate any signature present on the message. The signature
433 **MUST** be the signature of the <ProviderID> contained in the message. If the signature is
434 invalid, the identity provider **MUST** ignore the message.

- 435 **IDP-NIM-004** The identity provider **MUST** respond to the <lib:NameIdentifierMappingRequest>
436 with a SOAP 200 OK <lib:NameIdentifierMappingResponse> message. The identity
437 provider is **NOT** required to honor the request for a mapped NameIdentifier, but it **MUST** re-
438 spond to the request with an appropriate status.

- 439 **IDP-NIM-005** The identity provider **MUST** construct the <lib:NameIdentifierMappingResponse> in
440 accordance with the rules described in [[LibertyProtSchema, 3.6.3.1](#)].

- 441 **IDP-NIM-006** The identity provider **SHOULD** encrypt or otherwise obfuscate the NameIdentifier returned to
442 the requesting service provider.

443 **2.3.4.1. Encrypted Name Identifiers**

444 This section describes the requirements an identity provider must meet when encrypting and encoding a NameIdentifier,
445 such as might be needed to satisfy [[IDP-NIM-006](#)].

- 446 **IDP-ENI-001** Encrypted Name Identifiers are a mandatory supported feature of the Extended Identity Provider
447 Conformance Profile. The requirements in this section **MUST** be implemented according to
448 the relevant section of [[LibertyBindProf, 3.8](#)].

449 **IDP-ENI-002** This interaction MUST NOT be used unless the provider metadata element
450 NameIdentifierMappingEncryptionProfile specifies the URI
451 *urn:liberty:iff:nameid:encrypted*.

452 **2.3.4.1.1. XML Encrypting and Encoding**

453 The <EncryptableNameIdentifier> element is used to contain the encrypted data.

454 **IDP-ENI-003** The identity provider MUST transform the original <saml:NameIdentifier>element into a
455 <EncryptableNameIdentifier> element, encrypt and encode it as described in [[Liberty-](#)
456 [BindProf](#), 3.8.1.1].

457 The requirements in this section rely on the use of [[xmlesc-core](#)] to format and encode the encrypted data. Note that
458 certain block encryption algorithms are mandatory for use with the <xenc:EncryptedData> and therefore must be
459 supported:

460 **IDP-ENI-004** The following algorithms MUST be supported as indicated in [[xmlesc-core](#)], sections 5.2.1 and
461 5.2.2: TRIPLE DES, AES-128, AES-256.

462 **2.3.4.1.2. Security Considerations**

463 The security considerations surrounding the Encrypted NameIdentifier are described in [[LibertyBindProf](#), 3.8.2].

464 **IDP-ENI-005** The identity provider MUST follow the requirements in [[LibertyBindProf](#), 3.8.2] regarding the
465 handling of symmetric encryption keys.

466 **2.4. Service Provider Basic Conformance Profile**

467 This section defines the minimal conformance requirements for a Service Provider. These requirements are derived
468 from the steps defined in the [[LibertyBindProf](#), [Section 3](#)], which describe the interactions between the user agent, the
469 service provider, and the identity provider.

470 **2.4.1. Single Sign-On and Federation**

471 **2.4.1.1. Common Interaction and Processing**

472 The single sign-on requirements specified here assume that the user agent has already authenticated with the identity
473 provider, and that a valid session exists for the user agent at the identity provider.

474 **2.4.1.2. Single Sign-on using Browser Artifact**

475 This section describes the service provider actions necessary to perform single sign-on using browser artifact.

476 **SP-SSO-001** Single sign-on using browser artifact is a mandatory feature of the service provider basic
477 conformance profile. The requirements in this section MUST be implemented as specified
478 in [[LibertyBindProf](#), [Section 3.2.2](#)] and [[LibertyBindProf](#), 3.2.1, [Steps 8 and 10](#)].

479 The user agent initiates the single sign-on by making an HTTP request to the service provider, as indicated in
480 [[LibertyBindProf](#), 3.2.1.1, [Step 1](#)]. The service provider then obtains the address of the appropriate identity provider
481 in an implementation-dependent way (not normatively specified).

547 **SP-RNI-006** The identity provider will send a `<lib:RegisterNameIdentifierRequest>` pro-
548 tocol message to the service provider. The service provider **MUST** record the new
549 `<lib:IDPProvidedNameIdentifier>`.

550 **SP-RNI-007** After a successful registration of the `<lib:IDPProvidedNameIdentifier>`, the service
551 provider **MUST** respond with a `<lib:RegisterNameIdentifierResponse>` according to
552 the processing rules in [[LibertyProtSchema, 3.3.3](#)].

553 **2.4.2.2. Register Name Identifier Initiated at Service Provider**

554 Note that this section refers to [[LibertyBindProf](#)] register name identifier interactions initiated at the identity provider.
555 The steps associated with the service provider in those interactions are used here as if they were associated with the
556 identity provider. All references to service provider and identity provider have been interchanged as indicated in
557 [[LibertyBindProf, 3.3.2.2](#)].

558 **2.4.2.2.1. HTTP-Redirect Based**

559 **SP-RNI-008** The HTTP-Redirect based register name identifier (initiated at the service provider) is a
560 **REQUIRED** feature of the service provider basic profile.

561 The service provider can initiate the register name identifier interaction, though the circumstances of this initiation are
562 not normatively specified.

563 **SP-RNI-009** The service provider **MUST** redirect the user agent to the register name identifier service at the
564 identity provider as specified in [[LibertyBindProf, 3.3.1.1.2, Step 2](#)].

565 **2.4.2.2.2. SOAP/HTTP Based**

566 **SP-RNI-010** The SOAP/HTTP based register name identifier (initiated at the service provider) is an **OP-**
567 **TIONAL** feature of the service provider basic profile.

568 **SP-RNI-011** The service provider **MUST** only initiate SOAP/HTTP-based register name identifier when the
569 identity provider metadata specifies the appropriate URI identifier as specified in [[LibertyBind-](#)
570 [Prof, 3.3.1.2](#)].

571 **SP-RNI-012** The SOAP/HTTP-based register name identifier transactions **MUST** use the SOAP binding for
572 Liberty as defined in [[LibertyBindProf, 2.1](#)].

573 **SP-RNI-013** The service provider **MUST** initiate the register name identifier transaction by sending a
574 `<lib:RegisterNameIdentifierRequest>` message to the identity provider's SOAP end-
575 point as specified in [[LibertyBindProf, 3.3.1.2, Step 1](#)].

576 **SP-RNI-014** The service provider **MUST** process the `<lib:RegisterNameIdentifierResponse>` from
577 the identity provider as specified in [[LibertyProtSchema, 3.3.3](#)].

578 **2.4.3. Identity Federation Termination Notification**

579 Liberty identity federation termination notification specifies how service providers and identity providers are notified
580 of federation termination. There are four variations of the federation termination notification interaction: the
581 federation termination notification interaction can be initiated by either the identity provider or the service provider,
582 and the protocol can be based on either HTTP-Redirect or SOAP/HTTP.

583 **2.4.3.1. Federation Termination Notification Initiated at the Identity Provider**

584 **2.4.3.1.1. HTTP-Redirect**

585 **SP-FTN-001** The HTTP-Redirect based federation termination notification (initiated at the identity provider)
586 is a REQUIRED feature of the service provider basic profile.

587 **SP-FTN-002** The service provider MUST process the `<lib:FederationTerminationNotification>`
588 received from the user agent according to the rules defined in [[LibertyProtSchema, 3.4.2](#)] and
589 in [[LibertyBindProf, 3.4.1.1.4, Step 4](#)].

590 **SP-FTN-003** The service provider's federation termination service MUST respond by redirecting the user
591 agent as specified in [[LibertyBindProf, 3.4.1.1.5, Step 5](#)].

592 **2.4.3.1.2. SOAP/HTTP**

593 **SP-FTN-004** The SOAP/HTTP based federation termination notification (initiated at the identity provider) is
594 an OPTIONAL feature of the service provider basic profile.

595 **SP-FTN-005** The service provider MUST process the `<lib:FederationTerminationNotification>`
596 in the SOAP message received from the identity provider according to the rules defined in
597 [[LibertyProtSchema, 3.4.2](#)] and in [[LibertyBindProf, 3.4.1.2.3, Step 3](#)].

598 **SP-FTN-006** The service provider MUST respond to the `<lib:FederationTerminationNotification>`
599 with a HTTP 204 OK response [[LibertyBindProf, 3.4.1.2.4, Step 4](#)].

600 **2.4.3.2. Federation Termination Initiated at the Service Provider**

601 **2.4.3.2.1. HTTP-Redirect**

602 **SP-FTN-007** The HTTP-Redirect based federation termination notification (initiated at the service provider)
603 is a REQUIRED feature of the service provider basic profile.

604 **SP-FTN-008** This interaction MUST NOT be used unless the identity provider metadata ele-
605 ment `FederationTerminationNotificationProtocolProfile` specifies the URI
606 *<http://projectliberty.org/profiles/fedterm-sp-http>*.

607 **SP-FTN-009** This interaction REQUIRES certain preconditions specified in [[LibertyBindProf, 3.4.1.1](#)] are
608 met.

609 **SP-FTN-010** In response to a request to the service provider's federation termination service URL, the service
610 provider **MUST** redirect the user agent to the federation termination service at the identity
611 provider. This redirection **MUST** adhere to the rules specified in [[LibertyBindProf, 3.4.1.1.2](#),
612 Step 2].

613 **2.4.3.2.2. SOAP/HTTP**

614 **SP-FTN-011** The SOAP/HTTP based federation termination notification (initiated at the service provider) is
615 an **OPTIONAL** feature of the service provider basic profile.

616 **SP-FTN-012** This interaction **MUST NOT** be used unless the identity provider metadata ele-
617 ment `FederationTerminationNotificationProtocolProfile` specifies the URI
618 *<http://projectliberty.org/profiles/fedterm-sp-soap>*.

619 **SP-FTN-013** This interaction **REQUIRES** certain preconditions specified in [[LibertyBindProf, 3.4.1.2](#)] are
620 met.

621 **SP-FTN-014** In response to a request from the user agent to the service provider's federation termination
622 service URL, the service provider **MUST** send an asynchronous SOAP over HTTP notification
623 message to the identity provider's SOAP endpoint. The SOAP message **MUST** adhere to the
624 rules specified in [[LibertyBindProf, 3.4.1.2.2, Step 2](#)].

625 The identity provider will respond to termination notification with a HTTP 204 No Content response.

626 **SP-FTN-015** The service provider **MUST** process the HTTP 204 No Content response from the identity
627 provider and send an HTTP response confirming the requested action of federation termination
628 with the specified identity provider.

629 **2.4.4. Single Logout**

630 Liberty single logout specifies how service providers and identity providers synchronize logout across all sessions
631 authenticated by a particular identity provider. There are five variations of the single logout interaction: the single
632 logout can be initiated by either the identity provider or the service provider, and the protocol can be based on either
633 HTTP-Redirect, HTTP-GET (only when initiated at the identity provider), or SOAP/HTTP.

634 Note that Single Logout, in the general case, is an iterative process from the perspective of an identity provider since
635 the identity provider must contact each service provider to which it has issued authentication assertions. However,
636 for a service provider, the single logout interaction is a single event.

637 **2.4.4.1. Single Logout Initiated at the Identity Provider**

638 **2.4.4.1.1. HTTP-Redirect**

639 **SP-SLO-001** The HTTP-Redirect based single logout interaction (initiated at the identity provider) is a
640 **REQUIRED** feature of the service provider basic profile.

641 **SP-SLO-002** The user agent will access the service provider's single logout service URL via a redirect
642 from the identity provider. The service provider **MUST** process the `<lib:LogoutRequest>`
643 according to the rules defined in [[LibertyProtSchema, 3.5.1](#)]

644 **SP-SLO-003** The service provider **MUST** invalidate the session(s) of the Principal referred to in the name
645 identifier received from the identity provider in the `<lib:LogoutRequest>`.

646 **SP-SLO-004** The service provider **MUST** respond and redirect the user agent back to the identity provider us-
647 ing the return URL location obtained from the `SingleLogoutServiceReturnURL` metadata
648 element as specified in [[LibertyBindProf, 3.5.1.1.1.5, Step 5](#)].

649 **2.4.4.1.2. HTTP-GET**

650 **SP-SLO-005** The HTTP-GET based single logout interaction (initiated at the identity provider) is a RE-
651 QUIRED feature of the service provider basic profile.

652 The user agent will access the single logout service URL of the service provide as a result of an image tag load
653 generated by the identity provider.

654 **SP-SLO-006** The service provider **MUST** process the `<lib:LogoutRequest>` according to the rules
655 defined in [[LibertyProtSchema, 3.5.1](#)]

656 **SP-SLO-007** The service provider **MUST** invalidate the session(s) of the Principal referred to in the name
657 identifier received from the identity provider in the `<lib:LogoutRequest>`.

658 **SP-SLO-008** The service provider **MUST** respond and redirect the user agent image load back to the identity
659 provider's logout completion URL obtained from the `SingleLogoutServiceReturnURL`
660 metadata element. The HTTP response **MUST** be formed as specified in [[LibertyBindProf,](#)
661 [3.5.1.1.2.5, Step 5](#)].

662 **2.4.4.1.3. SOAP/HTTP**

663 **SP-SLO-009** The SOAP/HTTP based single logout interaction (initiated at the identity provider) is an
664 OPTIONAL feature of the service provider basic profile.

665 **SP-SLO-010** After receiving a `<lib:LogoutRequest>` from the identity provider, the service provider
666 **MUST** process it according to the rules in [[LibertyProtSchema, 3.5.1](#)].

667 **SP-SLO-011** The service provider **MUST** invalidate the session(s) of the Principal referred to in the name
668 identifier received from the identity provider in the `<lib:LogoutRequest>`.

669 **SP-SLO-012** The service provider **MUST** respond to the `<lib:LogoutRequest>` with a SOAP 200 OK
670 `<lib:LogoutResponse>` message [[LibertyBindProf, 3.5.1.2, Step 4](#)].

671 2.4.4.2. Single Logout Initiated at the Service Provider

672 2.4.4.2.1. HTTP-Redirect

673 **SP-SLO-013** The HTTP-Redirect based single logout interaction (initiated at the service provider) is a
674 REQUIRED feature of the service provider basic profile.

675 The user agent will access the single logout service URL at the service provider.

676 **SP-SLO-014** The service provider's single logout service responds and redirects the user agent to the single
677 logout service at the identity provider. The HTTP redirect MUST adhere to the rules specified
678 in [[LibertyBindProf, 3.5.2.1.2, Step 2](#)].

679 After the identity provider has processed the single logout request and contacted the appropriate service providers, the
680 user agent will be redirected back to the service provider contacted originally.

681 **SP-SLO-015** The service provider SHOULD send an HTTP 200 OK response to the user agent with
682 confirmation of the logout.

683 2.4.4.2.2. SOAP/HTTP

684 **SP-SLO-016** The SOAP/HTTP based single logout interaction (initiated at the service provider) is an
685 OPTIONAL feature of the service provider basic profile.

686 The user agent will initiate single logout by accessing the single logout service URL at the service provider via an
687 HTTP request.

688 **SP-SLO-017** In response to the single logout request, the service provider sends a SOAP over HTTP request
689 to the identity provider's SOAP endpoint. The SOAP request MUST be constructed and
690 processed as specified in [[LibertyBindProf, 3.5.2.2, Step 2](#)].

691 The identity provider will contact all service providers to which it has issued assertions for the Principal to request a
692 logout action. The identity provider may determine that one or more of the service providers do not support the SOAP
693 single logout interaction. The identity provider will return a `<lib:LogoutResponse>` containing a status code of
694 `<lib:UnsupportedProfile>`

695 **SP-SLO-018** If the identity provider responds to the single logout request with `<lib:UnsupportedProfile>`,
696 the service provider MUST re-submit its `<lib:LogoutRequest>` via the HTTP interaction
697 specified above.

698 **SP-SLO-019** If the identity provider responds to the logout request with a SOAP 200 OK
699 `<lib:LogoutResponse>`, indicating successful single logout, the service provider SHOULD
700 send a HTTP response to the user agent confirming the single logout.

701 2.4.5. Identity Provider Introduction

702 This section describes the conformance requirements for a service provider implementing the identity provider
703 introduction feature. The identity provider introduction feature is intended to allow service providers to discover
704 which identity providers a Principal is using.

705 **SP-IPI-001** The identity provider introduction feature is an OPTIONAL element of the service provider
706 basic conformance profile.

707 **2.4.5.1. Common Domain Cookie**

708 The identity provider introduction relies on the use of a common domain cookie.

709 **SP-IPI-002** The common domain cookie **MUST** be constructed as specified in [[LibertyBindProf, 3.6.1](#)].

710 **2.4.5.2. Obtaining the Common Domain Cookie**

711 The service provider uses the common domain cookie to determine which identity providers a Principal uses. The
712 common domain cookie is presented to the service provider after being read by an HTTP server in the common
713 domain; the details of this interaction are outside the scope of this document.

714 **SP-IPI-003** If the HTTP server in the common domain is operated by the service provider, the service
715 provider **MAY** redirect the user agent to an identity provider for single sign-on.

716 The details of this procedure are implementation-dependent, and are not normatively specified. However, one possible
717 strategy is described in [[LibertyBindProf, 3.6.3](#)].

718 **2.4.6. Backward Compatibility**

719 Backward compatibility is optional for conformant service providers:

720 **SP-BCK-001** Backward compatibility with 1.1 identity provider implementations is **OPTIONAL** for an
721 service provider implementation. However, an implementation claiming backward compatible
722 conformance **MUST** adhere to the relevant rules as described in [[LibertyProtSchema, 3.1.11](#)].

723 However, note that only required conformance features need interoperate:

724 **SP-BCK-002** Conformant service provider implementations are **NOT REQUIRED** to be backward compatible
725 with conformance features described as optional.

726 In other words, Service Provider Basic implementations need only be backward compatible for protocol features that
727 are required as part of the Service Provider Basic conformance profile.

728 **2.5. Service Provider Conformance Profile**

729 This section defines the conformance requirements for the Service Provider Conformance Profile.

730 **SP-CONF** The service provider profile is defined to be the service provider basic profile with all optional
731 interactions changed to **REQUIRED** except for the identity provider introduction interaction.

732 **2.6. Extended Service Provider Conformance Profile**

733 This section describes the requirements for an implementation to be an Extended Service Provider. This profile
734 extends the Service Provider Profile defined in the previous section.

735 **SP-EXT-001** An Extended Service Provider implementation **MUST** conform to the Service Provider profile
736 requirements. In addition, an Extended Service Provider **MUST** conform to all the require-
737 ments defined in the following subsections.

738 2.6.1. One-Time (Anonymous) Name Identifier

739 A service provider can request a one-time (also known as "anonymous") name identifier for a a Principal during the
740 sign-on interaction with the identity provider.

741 **SP-ONE-001** To request a one-time name identifier a service provider MUST specify a value of *onetime* for
742 the `NameIDPolicy` element in the `AuthnRequest`.

743 2.6.2. Affiliations

744 An affiliation is a set of one or more entities, described by `providerIDs`, who may perform Liberty interactions as a
745 member of the set. This section describes the conformance requirements for an service provider implementing the
746 affiliation features of the IDFF 1.2 specifications. These requirements are derived from protocol processing rules in
747 [[LibertyProtSchema](#)].

748 **SP-AFF-001** The value of the `<AffiliationID>` element of ID-FF 1.2 protocol messages MUST adhere to
749 the uniqueness constraints described in [[LibertyProtSchema, 3.1.3](#)].

750 2.6.2.1. Single Sign-on and `<AuthnRequest>`

751 The service provider will issue an `<AuthnRequest>` to initiate single sign-on. These are the requirements that obtain
752 when an affiliation is present in the request (see [[LibertyProtSchema, 3.2.2.6](#)]).

753 **SP-AFF-002** If the `<AffiliationID>` element is present, then the `<saml:NameIdentifier>` MUST be
754 the most recent name identifier provided by a member of the affiliation, if any, or the name
755 identifier for the Principal supplied by the identity provider for the affiliation.

756 **SP-AFF-003** `<AffiliationID>`, if present, MUST be the unique identifier of a known affiliation group
757 with which the identity provider has an established relationship, and of which the requesting
758 provider is a member.

759 2.6.2.2. Name Registration

760 The provider is required to correctly identify the affiliation in the `<RegisterNameIdentifierRequest>` and
761 `<RegisterNameIdentifierResponse>` messages.

762 **SP-AFF-004** In all of the name identifier elements in the request and response messages of this protocol,
763 if the Principal's identity federation is between the identity provider and an affiliation group
764 in which the service provider is a member, then the `NameQualifier` attribute MUST contain
765 the unique identifier of the affiliation group. Otherwise, it MUST contain the unique identifier
766 of the service provider. This attribute MUST be used by the providers to identify the specific
767 identity federation to be modified.

768 See [[LibertyProtSchema, 3.3.3](#)].

769 **2.6.2.3. Federation Termination**

770 The provider is required to correctly identify the affiliation when performing federation termination using the
771 <FederationTerminationNotification> message.

772 **SP-AFF-005** If the Principal's identity federation was between the identity provider and an affiliation group
773 in which the service provider is a member, then the `NameQualifier` attribute **MUST** contain
774 the unique identifier of the affiliation group. Otherwise, it **MUST** contain the unique identifier
775 of the service provider. This attribute **MUST** be used by the providers to identify the specific
776 identity federation being terminated.

777 See [[LibertyProtSchema, 3.4.2](#)].

778 **2.6.2.4. Single Logout**

779 The provider is required to correctly identify the affiliation when responding to a single logout request using the
780 <LogoutResponse> message.

781 **SP-AFF-006** If the Principal's identity federation is between the identity provider and an affiliation group
782 in which the service provider is a member, then the `NameQualifier` attribute **MUST** contain
783 the unique identifier of the affiliation group. Otherwise, it **MUST** contain the unique identifier
784 of the service provider. This attribute **MUST** be used by the providers to identify the specific
785 identity federation of the Principal who is logging out.

786 See [[LibertyProtSchema, 3.5.2.3](#)].

787 **2.6.3. Dynamic Proxy of Identity Provider**

788 A service provider may request that an identity provider obtain authentication from a third-party identity provider with
789 which a Principal has already authenticated.

790 The originator of an authentication request may control proxy behavior by including a <Scoping> element where
791 the provider sets a desired <ProxyCount> value and/or indicates a list of preferred identity providers which may be
792 proxied by defining an ordered <IDPList> of preferred providers.

793 **SP-PXY-001** A service provider **MUST** be able to construct an <AuthnRequest> with valid <Scoping>,
794 <ProxyCount>, and <IDPList> elements.

795 **2.6.4. Name Identifier Mapping**

796 This section describes the requirements for a service provider implementing the Name Identifier Mapping interaction.

797 **SP-NIM-001** The Name Identifier Mapping interaction is a **REQUIRED** feature of the Extended Service
798 Provider Conformance Profile.

799 A service provider may play one of two distinct roles in the NameIdentifier Mapping interaction. The first role is the
800 "Requester" which initiates the NameIdentifier request to the identity provider. The second role is the "Target"; the
801 Requester service provider may use the mapped NameIdentifier to issue a <saml:AttributeQuery> to the Target
802 to obtain additional information about the Principal.

803 **SP-NIM-002** This interaction MUST NOT be used unless the identity provider meta-
804 data element `NameIdentifierMappingProtocolProfile` specifies the URI
805 *http://projectliberty.org/profiles/nim-sp-http*.

806 **2.6.4.1. Requester Role**

807 The steps performed by the service provider in the Requester role are illustrated in [[LibertyBindProf, Figure 17](#)]. The
808 service provider initiates the NameIdentifier Mapping interaction by sending a SOAP-over-HTTP request to the SOAP
809 endpoint of the identity provider it is querying.

810 **SP-NIM-003** The SOAP message MUST contain exactly one `<lib:NameIdentifierMappingRequest>`
811 element in the SOAP body, and follow the construction rules defined in [[LibertyProtSchema,](#)
812 3.6.1.1]. In addition, the message MUST be signed.

813 **SP-NIM-004** If the Principal's identity federation is between the identity provider and an affiliation group
814 in which the Requester service provider is a member, the `NameQualifier` attribute of the of
815 the request message's `<saml:NameIdentifier>` MUST contain the unique identifier of the
816 affiliation group. Otherwise, it MUST contain the unique identifier of the service provider.
817 See [[LibertyProtSchema, 3.6.3](#)].

818 The identity provider will respond to the `<lib:NameIdentifierMappingRequest>` with a SOAP 200 OK
819 `<lib:NameIdentifierMappingResponse>` message.

820 **SP-NIM-005** The service provider MUST validate any signature on the response message. The signature on
821 the message MUST be the signature of the `<ProviderID>` contained in the message. If the
822 signature is not valid, the service provider MUST ignore the message.

823 The processing rules for the Requester service provider after receiving and validating the
824 `<lib:NameIdentifierMappingResponse>` are not normatively specified by Liberty. See [[LibertyBindProf,](#)
825 3.7.1.4 and 3.7.1.5] for more information.

826 **2.6.4.2. Target Role**

827 The actions of a service provider in an Target role are not normatively specified by Liberty. See [[LibertyBindProf,](#)
828 3.7.1.4 and 3.7.1.5] for more information.

829 **2.6.4.3. Encrypted Name Identifiers**

830 This section describes the requirements a service provider must meet when decoding and decrypting a NameIdentifier
831 value such as one that might be obtained from an identity provider in the context of [[IDP-NIM-006](#)]. Note that in this
832 context, the service provider is playing the role of target (see [[role definition](#)]).

833 **SP-ENI-001** This interaction MUST NOT be used unless the provider metadata element
834 `NameIdentifierMappingEncryptionProfile` specifies the URI
835 *urn:liberty:iff:nameid:encrypted*.

836 **2.6.4.3.1. Decoding and Decrypting**

837 The processing steps for a provider receiving such an encrypted NameIdentifier are not normatively specified by
838 Liberty, but are described in [\[LibertyBindProf, 3.8.1.2\]](#).

839 Note that certain block encryption algorithms are mandatory for use with the `<xenc:EncryptedData>` and therefore
840 must be supported:

841 **SP-ENI-002** The following algorithms MUST be supported as indicated in [\[xmenc-core\]](#), sections 5.2.1 and
842 5.2.2: TRIPLE DES, AES-128, AES-256.

843 **2.6.4.3.2. Security Considerations**

844 **SP-ENI-003** The provider receiving an encrypted NameIdentifier MUST take care to use the `IssueInstant`
845 and `Nonce` attributes to prevent replay and long-term use of the same encrypted identifier.

846 **2.7. Liberty Enabled Client/Proxy (LECP)**

847 This section contains detailed specifications of the LECP Profile.

848 **2.7.1. General LECP Requirements**

849 **LCP-SSO-001** All HTTP requests made by a LECP MUST include a Liberty-Enabled indication. A Liberty-
850 Enabled indication is either a Liberty-Enabled header or User-Agent header containing a
851 Liberty-Enabled value as defined in [\[LibertyBindProf, 3.2.4.1\]](#).

852 The preferred Liberty-Enabled indication is the Liberty-Enabled header.

853 **LCP-SSO-002** A LECP SHOULD add the Liberty-Enabled header to each HTTP request. This header MUST
854 be constructed as specified in [\[LibertyBindProf, 3.2.4.1\]](#).

855 **LCP-SSO-003** A LECP MAY add a Liberty-Enabled entry in the HTTP User-Agent request header, as specified
856 in [\[LibertyBindProf, 3.2.4.1\]](#)

857 **2.7.2. Single Sign-On**

858 The single sign-on interaction is the only LECP interaction specified. This interaction assumes that the user agent has
859 authenticated at the identity provider and that a valid session exists for the user agent at the identity provider.

860 **LCP-SSO-004** To initiate single sign-on, the user agent MUST contain at most one Liberty-Enabled header.
861 If a proxy receives a HTTP request that contains a Liberty-Enabled header, it MUST NOT add
862 another Liberty-Enabled header.

863 **LCP-SSO-005** A proxy MAY replace the Liberty-Enabled header, but this replacement MUST adhere to the
864 specifications in [\[LibertyBindProf, 3.2.4.2, Step 1\]](#)

865 After receiving the HTTP 200 OK response (containing a `<lib:AuthnRequestEnvelope>` from the service provider,
866 the LECP will determine the correct identity provider to use.

- 867 **LCP-SSO-006** The LECP MUST issue an HTTP POST of the `<lib:AuthnRequest>` in the body
868 of a SOAP message to the identity provider's single sign-on service URL. This
869 MUST be the same `<lib:AuthnRequest>` as was received in the service provider's
870 `<lib:AuthnRequestEnvelope>`. See [[LibertyBindProf, 3.2.4.2, Step 4](#)].
- 871 **LCP-SSO-007** In case of any error, the LECP MUST return a `<lib:AuthnResponse>` to the service provider
872 as specified in [[LibertyBindProf, 3.2.4.2, Step 4](#)].
- 873 After receiving a HTTP response from the identity provider, the LECP will issue an HTTP POST to the service
874 provider.
- 875 **LCP-SSO-008** The HTTP POST from the LECP MUST be composed as specified in [[LibertyBindProf, 3.2.4.2,](#)
876 Step 7].
- 877 **LCP-SSO-009** In case of any error, the LECP MUST return a `<lib:AuthnResponse>` to the service provider
878 as specified in [[LibertyBindProf, 3.2.4.2, Step 7](#)].

879 **References**

880 **Normative**

- 881 [LibertyProtSchema] Cantor, Scott, Kemp, John, eds. "Liberty ID-FF Protocols and Schema Specification," Version
882 1.2-errata-v3.0, Liberty Alliance Project (14 December 2004). <http://www.projectliberty.org/specs>
- 883 [LibertyBindProf] Cantor, Scott, Kemp, John, Champagne, Darryl, eds. "Liberty ID-FF Bindings and
884 Profiles Specification," Version 1.2-errata-v2.0, Liberty Alliance Project (12 September 2004).
885 <http://www.projectliberty.org/specs>
- 886 [LibertyMetadata] Davis, Peter, eds. "Liberty Metadata Description and Discovery Specification," Version 1.1, Liberty
887 Alliance Project (14 December 2004). <http://www.projectliberty.org/specs>
- 888 [SAMLCore11] Maler, Eve, Mishra, Prateek, Philpott, Rob, eds. (27 May 2003). "Assertions and Protocol
889 for the OASIS Security Assertion Markup Language (SAML) V1.1," OASIS Committee Specification,
890 version 1.1, Organization for the Advancement of Structured Information Standards [http://www.oasis-](http://www.oasis-open.org/committees/documents.php?wg_abbrev=security)
891 [open.org/committees/documents.php?wg_abbrev=security](http://www.oasis-open.org/committees/documents.php?wg_abbrev=security)
- 892 [RFC2119] Bradner, S., eds. "Key words for use in RFCs to Indicate Requirement Levels," RFC 2119, The Internet
893 Engineering Task Force (March 1997). <http://www.ietf.org/rfc/rfc2119.txt> [March 1997].
- 894 [xmlenc-core] Eastlake, Donald, Reagle, Joseph, eds. (December 2002). "XML Encryption Syntax and Processing,"
895 W3C Recommendation, World Wide Web Consortium <http://www.w3.org/TR/xmlenc-core/>

896 **Informative**

- 897 [LibertyImplGuide] "Liberty ID-FF Implementation Guidelines," Version 1.2, Liberty Alliance Project (18 April
898 2004). <http://www.projectliberty.org/specs> Thompson, Peter, Champagne, Darryl, eds.
- 899 [LibertyGlossary] "Liberty Technical Glossary," Version 1.4, Liberty Alliance Project (14 Dec 2004).
900 <http://www.projectliberty.org/specs> Hodges, Jeff, eds.