

1 **OASIS SSTC SAML Protocols Schema**
2 **Discussion**

3
4 draft-sstc-protocol-discussion-00.doc

5
6 23 July 2001

7 Authors:
8 Chris McLaren, Netegrity
9 Prateek Mishra, Netegrity

10

10

11	OASIS SSTC SAML Protocols Schema Discussion	I
12	1 Overview	3
13	2 General Architecture	3
14	2.1 Discussion	3
15	2.1.1 Inheritance Scheme	3
16	2.1.1.1 SAMLAbsertRequest and SAMLRequest	3
17	2.1.2 Example: SAMLRequest	4
18	2.1.3 SAMLQuery and SubjectQuery	4
19	2.1.3.1 Concrete Query Types	5
20	2.2 Issues	5
21	2.2.1 ISSUE:[PRO-01] Return of expired assertions?	5
22	2.2.2 ISSUE:[PRO-02] Assertion Lookup via Other Identifiers	5
23	3 Authentication Query	5
24	3.1 Discussion	5
25	3.1.1 Example: <SAMLQuery> with type AuthenticationQueryType	6
26	4 Attribute Query	6
27	4.1 Discussion	6
28	4.2 Issues	7
29	4.2.1 ISSUE:[PRO-03] ANY/ALL Qualifier	7
30	4.2.2 ISSUE:[PRO-04] Returning Attribute Names Only	7
31	4.2.3 ISSUE:[PRO-05] ALL or ``Error''	8
32	4.2.4 ISSUE:[PRO-06] Expressiveness of Query Language	8
33	4.3 Example	8
34	5 Authorization Query	8
35	5.1 Discussion	8
36	5.2 Issues	9
37	5.3 Example	9
38	6 SAMLResponse and SAMLAbsertResponseType	10
39	6.1 Discussion	10
40	6.1.1 SAMLAbsertResponseType	10
41	6.1.2 SAMLResponseType	10
42	6.2 Issues	11
43	6.2.1 ISSUE:[PRO-07 Status Code]	11
44	6.3 Examples	11
45	7 References	12
46		

47

47 1 Overview

48 This document is the second document in a series of documents which provide a commentary on
 49 the SAML schema [draft-schema-assertion-10] and [draft-schema-protocols-10]. The first
 50 document [draft-core-discussion-00] describes the general principles governing the SAML
 51 schema design and comments on the assertion schema.

52 2 General Architecture

53 2.1 Discussion

54 The SAML Protocols schema describes a request-response protocol for three different types of
 55 queries. Our architecture is based one request type, and one response type for all of these queries.
 56 The request type is capable of carrying many different types of queries all of which extend from
 57 one of two common abstract extension points. Three such concrete extensions are defined in the
 58 specification.

59 2.1.1 Inheritance Scheme

60 In order to ask a question in SAML one creates an instance of the SAMLRequest element. The
 61 SAMLRequest element's type definition is an extension of SAMLAbstractRequestType, which
 62 is provided as an extension point for potential future requirements.

63 On of possible content element in a SAMLRequest is a query. This query must be a descendant
 64 of the abstract SAMLQueryType. Three different such descendants are defined in the
 65 specification: AuthenticationQueryType, AttributeQueryType, and AuthorizationQueryType.
 66 Note that all of these are descended from the abstract base type SubjectQuery, which is in turn
 67 descended from the SAMLQueryType.

68 2.1.1.1 SAMLAbstractRequest and SAMLRequest

```

69 <xsd:complexType name="SAMLAbstractRequestType" abstract="true">
70   <xsd:attribute name="RequestID" type="saml:IDType" use="required"/>
71   <xsd:attribute name="Version" type="string" use="required"/>
72 </xsd:complexType>
73
74 <element name ="SAMLRequestType" type="samlp:SAMLRequestType">
75 <complexType name="SAMLRequestType">
76   <complexContent>
77     <extension base="samlp:SAMLAbstractRequestType">
78       <choice>
79         <element ref="saml:SAMLQuery"/>
80         <element ref="saml:AssertionID"
81           maxOccurs="unbounded"/>
82       </choice>
83     </extension>
84   </complexContent>

```

85 </complexType>

86
 87 The SAMLAbstractRequest defines only two attributes, both of which are required. It is an
 88 abstract type that can not be instantiated and is only provided as an extension point for possible
 89 future uses.

90 **RequestID:** This is a unique identifier for the request (See also ISSUE:[CONS-02])

91 **Version:** This is the string that identifies the version of the SAML protocol being used (See also
 92 ISSUE:[CONS-05]).

93 The SAMLRequestType extends the abstract base by adding one of two elements to its body.
 94 The element <SAMLRequest> is made available as a container for SAMLRequestType objects.

95 The element found within <SAMLRequest> indicates the type of the question asked:

96 If an <AssertionID> is present in the body the question is interpreted as “Would you return the
 97 Assertion corresponding to this AssertionID?”

98 Otherwise some descendant of the abstract SAMLQueryType will be present, and will contain a
 99 more specific question. These specific questions are detailed in §4 through §6.

100 **2.1.2 Example: SAMLRequest**

```
101 <SAMLRequest RequestID="urn:SAMLID:17862301" Version="100">
102   <AssertionID>urn:SAMLID:29100231XA</AssertionID>
103 </SAMLRequest>
```

104

105 In this example, the <SAMLRequest> element is requesting a SAML assertion with a certain
 106 <AssertionID>. If successful, the assertion will be returned within a <SAMLResponse> element.

107 **2.1.3 SAMLQuery and SubjectQuery**

```
108 <complexType name="SAMLQueryType" abstract="true"/>
109
110 <complexType name="SubjectQueryType" abstract="true">
111   <complexContent>
112     <extension base="samlp:SAMLQueryType">
113       <sequence>
114         <element ref="saml:Subject"/>
115       </sequence>
116     </extension>
117   </complexContent>
118 </complexType>
```

119

120 The SAMLQuery element is an abstract element that exists to provide both an extension point
 121 for possible future query types and could also act as the head element of a substitution group that
 122 covers all possible query types. The type has no defined characteristics.

123 The SubjectQueryType is a second abstract type, derived from the SAMLQueryType with the
124 addition of one and only one <Subject> element. For information on the <Subject> element see
125 §4.1.3.1 of draft-sstc-core-XX-discussion.doc

126 The SubjectQueryType is the base class from which the three concrete query classes defined in
127 this specification are derived.

128 **2.1.3.1 Concrete Query Types**

129 Three concrete query types are defined in this specification: an authentication query, an attribute
130 query, and an authorization query. Depending on what sort of question a SAMLRequest is
131 contains, one of these types is used as the SAMLQuery descendant within the request. The query
132 types defined in this specification are detailed in §4 through §6.

133 **2.2 Issues**

134 **2.2.1 ISSUE:[PRO-01] Return of expired assertions?**

135 Does the specification make any normative statements about the expiry state of assertions
136 returned in response to SAMLRequests? Is it a requirement that only unexpired assertions are
137 returned, or is the client responsible for checking? (*Seems pretty clear that the client will have to
138 check anyway at time-of-use, so forcing the responder to check before replying seems like extra
139 processing.*)

140 **2.2.2 ISSUE:[PRO-02] Assertion Lookup via Other Identifiers**

141 In some instances (such as the web browser profile) it is necessary to lookup an assertion using
142 an identifier other than the <AssertionID>. Typically, such an identifier is opaque and may have
143 been created in some proprietary way by an asserting party. Do we need an additional element in
144 SAMLRequestType to model this type of lookup?

145 **3 Authentication Query**

146 **3.1 Discussion**

```
147 <complexType name="AuthenticationQueryType">
148     <complexContent>
149         <extension base="samlp:SubjectQueryType">
150             <sequence>
151                 <element ref="saml:AuthenticationCode"
152                         minOccurs="0"/>
153             </sequence>
154         </extension>
155     </complexContent>
156 </complexType>
```

157

158 The AuthenticationQueryType asks the question “What authentication assertions are available
 159 for this Subject?”
 160 An AuthenticationQuery contains all the elements and attributes of a SubjectQuery and extends
 161 them with the optional addition of an AuthenticationCode. The <AuthenticationCode> element is
 162 defined in §4.1.4 of draft-sstc-core-XX-discussion.doc. In the context of a query this code can be
 163 used to as a filter for possible responses. This supports the query “What authentication assertions
 164 do you have for this Subject with the following AuthenticationCode?”
 165 A SAML processor will return a certain number of AuthenticationAssertions in response to a
 166 SAMLQuery of type AuthenticationQueryType. The <Subject> and <AuthenticationCode> of
 167 the returned assertions must be identical to the subject (and optional <AuthenticationCode>)
 168 fields of the SAMLQuery. There is no implication that all such assertions must be returned.

169 ***3.1.1 Example: <SAMLQuery> with type AuthenticationQueryType***

```
170 <SAMLQuery xsi:type="samlp:AuthenticationQueryType" >
171   <Subject>
172     <NameIdentifier>
173       <SecurityDomain>www.example.com</SecurityDomain>
174       <Name>cn=SomeUser,co=example,ou=sales</Name>
175     </NameIdentifier>
176   </Subject>
177   <AuthenticationCode>X509v3</AuthenticationCode>
178 </AuthenticationQuery>
```

179

180 In this example we are asking for all authentication assertions for a subject with “cn=SomeUser,
 181 co=example, ou=sales” with <AuthenticationCode> set to X509v3.

182 When wrapped in a <SAMLRequest> element we have the complete query:

```
183 <SAMLRequest RequestID="{EE52CAF4-3768-4ebe-84D3-4D372C892A5D}">
184   Version="http://www.oasis.org/tbs/1066-12-25/protocol/">
185     <SAMLQuery xsi:type="samlp:AuthenticationQueryType" >
186       <Subject>
187         <NameIdentifier>
188           <SecurityDomain>www.example.com</SecurityDomain>
189           <Name>SomeUser</Name>
190         </NameIdentifier>
191       </Subject>
192       <AuthenticationCode>X509v3</AuthenticationCode>
193     </SAMLQuery>
194   </SAMLRequest>
```

195

196 **4 Attribute Query**

197 **4.1 Discussion**

```
198 <complexType name="AttributeQueryType" >
```

```
199      <complexContent>
200          <extension base="samlp:SubjectQueryType">
201              <sequence>
202                  <element ref="saml:Attribute" minOccurs="0"
203 maxOccurs="unbounded"/>
204                  <element name="CompletenessSpecifier"
205                      type="samlp:CompletenessSpecifierType"
206                      default="All"/>
207              </sequence>
208          </extension>
209      </complexContent>
210  </complexType>
```

212 The AttributeQueryType can be used to ask the questions:

- “Return all of the attributes for this Subject (that I am allowed to see)?” The response will be in the form of an Attribute assertion.
- “Return ALL of the following attributes for this Subject?” The response will be in the form an Attribute assertion. If the requestor is not allowed access to ALL the requested attributes he will get NONE of them.
- “Return me ANY of the following attributes for this Subject?” The response will be in the form an Attribute assertion. Any attributes which the requestor is not allowed access to will not be present in the returned assertion.

221 An AttributeQuery contains all the elements and attributes of a SubjectQuery and extends them
222 with the addition of zero or more Attributes and exactly one CompletenessSpecifier.

223 The set of Attributes are used to specify the list of attributes in the query in the second and third
224 forms of the question. If there are no Attributes provided the query is interpreted as being the
225 first question above and the CompletenessSpecifier is ignored.

226 The CompletenessSpecifier is a two-value enumeration of “Any” or “All”. The chosen value
227 selects between the second and third questions listed above.

228 **4.2 Issues**

229 ***4.2.1 ISSUE:[PRO-03] ANY/ALL Qualifier***

230 Are these qualifiers necessary? Can we remove them?

231 See also ISSUE:[F2F#3-20, 3-21].

232 ***4.2.2 ISSUE:[PRO-04] Returning Attribute Names Only***

233 Do we need an additional query format where only the attribute names are returned? For
234 example, the query where all the attribute names for a subject are returned.

235 See also ISSUE:[F2F#3-22, 23].

236 **4.2.3 ISSUE:[PRO-05] ALL or ``Error''**

237 Is the all or “error” semantics for the ALL qualifier appropriate? Should we just follow LDAP
 238 semantics for this type of query?

239 See also [ISSUE:F2F#3-24] and discussion [F3] from [f2f#3-minutes]

240 **4.2.4 ISSUE:[PRO-06] Expressiveness of Query Language**

241 The approach taken in this draft is to use a very simple query language for attributes, essentially
 242 consisting of only of attribute names. At the f2f#3, there were suggestions that a richer query
 243 language might be used, such as for example based on XPATH. Is there any need to further
 244 explore this direction for SAML 1.0?

245 See also [F2] in [f2f#3-minutes]

246 **4.3 Example**

```

247 <SAMLQuery xsi:type="samlp:AttributeQueryType">
248   <Subject>
249     <NameIdentifier>
250       <SecurityDomain>www.example.com</SecurityDomain>
251       <Name>SomeUser</Name>
252     </NameIdentifier>
253   </Subject>
254   <CompletenessSpecifier>ANY</CompletenessSpecifier>
255   <Attribute>
256     <AttributeName>NetWorthSummary</AttributeName>
257     <AttributeNamespace>
258       http://ns.finance-vocab.org/finance
259     </AttributeNamespace>
260   </Attribute>
261   <Attribute>
262     <AttributeName>CreditScore</AttributeName>
263     <AttributeNamespace>
264       http://ns.finance-vocab.org/finance
265     </AttributeNamespace>
266   </Attribute>
267 </SAMLQuery>
```

268

269 This example is requesting two namespace-qualified attributes for the given Subject: a
 270 “NetWorthSummary” and a “CreditScore”. Since it is requested with the ANY flag that means
 271 the response will contain values for zero, one or both of these attribute names.

272 **5 Authorization Query**

273 **5.1 Discussion**

```

274 <complexType name="AuthorizationQueryType">
```

```

275     <complexContent>
276         <extension base="samlp:SubjectQueryType">
277             <sequence>
278                 <element ref="saml:Evidence" minOccurs="0"
279                         maxOccurs="unbounded"/>
280                 <element ref="saml:Object"/>
281             </sequence>
282         </extension>
283     </complexContent>
284 </complexType>

```

285

286 An Authorization Query is used to ask only one type of question: “Should action(s) Y on
 287 resource Z be allowed for subject S given evidence E?” The answer comes in the form of an
 288 Authorization Decision assertion. The action(s) and resource are optionally namespace-scoped
 289 and are contained with in the <Object> element of the query.

290 An AuthorizationQuery contains all the elements and attributes of a SubjectQuery and extends
 291 them with the addition of exactly one Object element and a collection zero or more assertions or
 292 assertionIDs to be used as evidence.

293 The Evidence and Object elements are defined in draft-sstc-core-XX-discussion.doc; Evidence in
 294 §4.1.6 and Object in §4.1.6.1.

295 5.2 Issues

296 5.3 Example

```

297 <SAMLQuery xsi:type="samlp:AuthorizationQueryType">
298     <Subject>
299         <NameIdentifier>
300             <SecurityDomain>us-staff</SecurityDomain>
301             <Name>cn=SomeUser,ou=finance,co=example</Name>
302         </NameIdentifier>
303     </Subject>
304     <Object>
305         <Resource>
306             http://www.example.com/confidential/agree.html
307         </Resource>
308         <Action>GET</Action>
309         <Action>POST</Action>
310         <Namespace>urn:samlaction:HTTP</Namespace>
311     </Object>
312     <Evidence>
313         <AssertionID>{EE52CAF4-3452-4ebe-84D3-4D372C892A5D}</AssertionID>
314     </Evidence>
315 </SAMLQuery>

```

316

317 In this example, the query concerns whether subject “cn=SomeUser,...” has the right to GET
 318 AND POST on a certain URL. The actions GET, POST are taken from the namespace
 319 urn:samlaction:HTTP.

320 6 SAMLResponse and 321 SAMLAbstractResponseType

322 6.1 Discussion

```

323 <complexType name="SAMLAbstractResponseType" abstract="true">
324   <attribute name="ResponseID" type="saml:IDType" use="required"/>
325   <attribute name="InResponseTo" type="saml:IDType" use="required"/>
326   <attribute name="Version" type="string" use="required"/>
327 </complexType>
328 <element name="SAMLResponse" type="SAMLResponseType" />
329 <complexType name="SAMLResponseType">
330   <complexContent>
331     <extension base="samlp:SAMLAbstractResponseType">
332       <sequence>
333         <element ref="saml:Assertion" minOccurs="0"
334           maxOccurs="unbounded" />
335       </sequence>
336       <attribute name="StatusCode" type="samlp:StatusCodeType"
337           use="required"/>
338     </extension>
339   </complexContent>
340 </complexType>

```

341
342 All types of requests in SAML are met with a common response, the <SAMLResponse>, which
343 has type SAMLResponseType. SAMLResponseType is an extension of the abstract base type
344 SAMLAbstractResponseType, which is provided as an extension point for other possible future
345 needs with respect to responses.

346 6.1.1 SAMLAbstractResponseType

347 The base type defines only the three required attributes of a response: the ResponseID, the
348 InResponseTo identifier, and the version string.

349 **RequestID:** This is the identifier for the response. See also [ISSUE:CONS-02]

350 **InResponseTo:** This attribute holds the identifier of the request which this response is
351 answering. Naturally whatever formatting rules, if any, that are arrived at for the request
352 identifiers will also apply to the type of this attribute.

353 **Version:** This is the string that identifies the version of the SAML protocol being used by the
354 responder. See also [ISSUE:CONS-05].

355 6.1.2 SAMLResponseType

356 The SAMLResponseType extends the SAMLAbstractResponseType with the addition of a
357 required Status Code attribute and a series of zero or more assertions that represent the answers
358 to the original request.

359 **StatusCode:** This attribute holds a string that describes the result of the original request, as
 360 expressed in this response. The contents of this attribute are restricted to members of an
 361 enumeration defined in the type “StatusCodeType” (see below).

```
362 <simpleType name="StatusCodeType">
363   <restriction base="string">
364     <enumeration value="Success"/>
365     <enumeration value="Failure"/>
366     <enumeration value="Error"/>
367     <enumeration value="Unknown"/>
368   </restriction>
369 </simpleType>
```

370

371 The assertions carried in the response represent the answers to the original question. The
 372 responding entity is responsible for what assertions are provided in response to any question.

373 6.2 Issues

374 6.2.1 ISSUE:[PRO-07 Status Code]

375 Are the status codes listed for StatusCodeType sufficient? If not how do we want to define a
 376 bigger list: keep it open with well-known values, use someone else’s list, define an extension
 377 system, etc.

378 See also ISSUE:[F2F#3-33, 34].

379 6.3 Examples

380 The example below shows an example of a successful response to the AttributeQuery shown in
 381 Section 5.3.

```
382 <SAMLResponse ResponseID="{'EF52CBF4-3452-4ebe-84D3-4D372C892A5D}'"
383   InResponseTo= "{EE52CAF4-3768-4ebe-84D3-4D372C892A5D}"
384   Version="0100"
385   StatusCode="Success">
386     <Assertion xsi:type="saml:AttributeAssertionType"
387       version="http://www.oasis.org/tbs/1066-12-25/"
388       AssertionID="{EE52CAF4-3452-4ebe-84D3-4D372C892A5D}"
389       Issuer="www.example.com"
390       IssueInstant="2001-05-31T13:20:00-05:00">
391         <Conditions NotBefore="2001-05-31T13:20:00-05:00"
392           NotOnOrAfter="2001-05-31T13:25:00-05:00" />
393         <Subject>
394           <NameIdentifier>
395             <SecurityDomain>www.example.com</SecurityDomain>
396             <Name>SomeUser</Name>
397           </NameIdentifier>
398         </Subject>
399         <Attribute>
400           <AttributeName>NetWorthSummary</AttributeName>
401           <AttributeNamespace>
```

```

402                               http://ns.finance-vocab.org/finance
403                         </AttributeNamespace>
404                         <AttributeValue>
405                           <CreditSummary>
406                             <HistoryScore>Excellent</HistoryScore>
407                             <CurrentAssets>Loaded</CurrentAssets>
408                           </CreditSummary>
409                         </AttributeValue>
410                       </Attribute>
411                     </Assertion>
412                   </SAMLResponse>

```

413

414 The example below shows an example of a successful response to the AuthorizationQuery
 415 shown in Section 6.3.

416

```

417 <SAMLResponse ResponseID="{'EF52CBF4-3452-4ebe-84D3-4D372C892A5D}'"
418   InResponseTo= "{EE52CAF4-3768-4ebe-84D3-4D372C892A5D}"
419   Version="0100"
420   StatusCode="Success">
421     <Assertion xsi:type="saml:AuthorizationDecisionAssertionType"
422       version="http://www.oasis.org/tbs/1066-12-25/"
423       AssertionID="{EE52CAF4-3452-4ebe-84D3-4D372C892A5D}"
424       Issuer="www.example.com"
425       IssueInstant="2001-05-31T13:20:00-05:00">
426         <Conditions NotBefore="2001-05-31T13:20:00-05:00"
427           NotOnOrAfter="2001-05-31T13:25:00-05:00" />
428         <Subject>
429           <NameIdentifier>
430             <SecurityDomain>us-staff</SecurityDomain>
431             <Name>cn=SomeUser,ou=finance,co=example</Name>
432           </NameIdentifier>
433         </Subject>
434         <Object>
435           <Resource>
436             http://www.example.com/confidential/agree.html
437           </Resource>
438           <Action>GET</Action>
439           <Action>POST</Action>
440           <Namespace>urn:samlaction:HTTP</Namespace>
441         </Object>
442         <Evidence>
443           <AssertionID>{EE52CAF4-3452-4ebe-84D3-4D372C892A5D}</AssertionID>
444         </Evidence>
445         <Answer>Permit</Answer>
446       </Assertion>
447     </SAMLResponse>

```

448

7 References

449 [f2f#3-minutes] <http://lists.oasis-open.org/archives/security-services/200107/msg00016.html>

450 [draft-sstc-core-discussion-00]