

2 **Security Assertions Markup Language**

3 *Core Assertion Architecture*

4 *Phillip Hallam-Baker*

VeriSign

5 *Tim Moses*

Entrust

6 *Bob Morgan*

University of Washington

7 *Carlisle Adams*

Entrust

8 *Charles Knouse*

Oblix

9 *David Orchard*

Jamcracker

10 *Eve Maler*

Sun

11 *Irving Reid*

Baltimore

12 *Jeff Hodges*

Oblix

13 *Marlena Erdos*

Tivoli

14 *Nigel Edwards*

Hewlett Packard

15 *Prateek Mishra*

Netegrity

16 *Chris McLaren*

Netegrity

17 *Draft Version 0.20: October 24th 2001*

19

20 Security Assertions Markup Language

21 Version 020

22 Table Of Contents

23	Table Of Contents	2
24	1 XML Assertion Syntax	4
25	1.1 Namespaces	5
26	1.1.1 Basic Types	5
27	1.2 SAML Assertion	7
28	1.2.1 Element <Assertion>	7
29	1.2.2 Element <SingleAssertion>	8
30	1.2.3 Element <MultipleAssertion>	8
31	1.2.4 Element <AssertionSpecifier>	9
32	1.2.5 Element <Statement>	9
33	1.3 Subject Statement	10
34	1.3.1 Element <SubjectStatement>	10
35	1.3.2 Element <Subject>	10
36	1.4 Authentication Assertion	12
37	1.4.1 Element <AuthenticationStatement>	12
38	1.5 Authorization Decision Statement	13
39	1.5.1 Element <AuthorizationStatement>	13
40	1.6 Attribute Statement	14
41	1.6.1 Element <AttributeStatement>	14
42	1.7 Conditions	16
43	1.7.1 Element <Conditions>	16
44	1.7.2 Condition Type AudienceRestrictionConditionType	17
45	1.8 Advice	18
46	1.8.1 Element <Advice>	18
47	2 SAML Protocol	20
48	2.1 Namespaces	20
49	2.1.1 Basic Types	20
50	2.2 Request	22
51	2.2.1 Abstract Type RequestAbstractType	22
52	2.2.2 Element <Request>	22
53	2.2.3 Element <Query>	23
54	2.2.4 Element <SubjectQuery>	23
55	2.3 Authentication Query	24
56	2.3.1 Element <AuthenticationQuery>	24
57	2.4 Attribute Query	24
58	2.4.1 Element <AttributeQuery>	24
59	2.5 Authorization Query	25

60	2.5.1	Element <AuthorizationQuery>	25
61	2.6	Response	26
62	2.6.1	Abstract Type ResponseAbstractType	26
63	2.6.2	Element <Response>	26
64	3	SAML Versions and Extension Schemas	28
65	3.1	SAML Version	28
66	3.1.1	Assertion Version	29
67	3.1.2	Request Version	29
68	3.1.3	Response Version	29
69	3.2	Schema Extension	29
70	3.2.1	Use of sub-typing and substitution groups	30
71	3.2.2	Extending the SAML Assertion Schema	30
72	3.2.3	Extending the SAML Protocol Schema	31
73	4	References	32
74	5	Identifiers	33
75	5.1	Authentication Protocol Identifiers	33
76	5.1.1	SAML Artifact	33
77	5.1.2	Assertion Bearer	33
78	5.1.3	User Name and Password (Pass-through)	33
79	5.1.4	User Name and Password (One-Way-Function SHA-1)	33
80	5.1.5	Kerberos	33
81	5.1.6	SSL/TLS Certificate Based Client Authentication	33
82	5.1.7	Object Authenticator (SHA-1)	33
83	5.2	Action Identifiers	33
84	5.2.1	Read/Write/Execute/Delete/Control	33
85	5.2.2	Read/Write/Execute/Delete/Control with Negation	33
86	5.2.3	Get/Head/Put/Post	33
87	5.2.4	UNIX file Permissions	33
88	6	Appendix	34
89	6.1	Assertion Schema	34
90	6.2	Protocol Schema	37
91			

92 1 XML Assertion Syntax

93 SAML specifies several different types of assertion for different purposes, these are:

94 **Authentication Assertion**

95 An authentication assertion contains a statement made by the issuer that asserts
96 the subject was authenticated by a particular means at a particular time.

97 **Authorization Decision Assertion**

98 An authorization decision assertion contains a statement made by the issuer that
99 asserts the request for access by the specified subject to the specified object has
100 resulted in the specified decision on the basis of some optionally specified
101 evidence.

102 **Attribute Assertion**

103 An attribute assertion contains a statement made by the issuer that asserts the
104 specified subject is associated with the specified attribute(s).

105 The different types of SAML assertion are encoded in a common XML package, which at
106 a minimum consists of:

107 **Basic Information.**

108 Each assertion **MUST** specify the version of the SAML assertion syntax, a unique
109 identifier that serves as a name for the assertion, a unique identifier for the issuer
110 and the time instant of issue.

111 **The Asserted Statement(s)**

112 The statement(s) that are asserted by the issuer of the assertion.

113 In addition an assertion **MAY** contain the following additional elements. An SAML
114 client is not required to support processing of any element contained in an additional
115 element **with the sole exception that an SAML client **MUST** reject any assertion
116 containing a **Conditions** element that is not supported.**

117 **Conditions.**

118 The assertion status **MAY** be subject to conditions. The status of the assertion
119 might be dependent on additional information from a validation service. The
120 assertion may be dependent on other assertions being valid. The assertion may
121 only be valid if the relying party is a member of a particular audience.

122 **Advice.**

123 Assertions **MAY** contain additional information as advice. The advice element
124 **MAY** be used to specify the assertions that were used to make a policy decision.

125 The SAML assertion package is designed to facilitate reuse in other specifications. For
126 this reason XML elements specific to the management of authentication and

127 authorization data are expressed as statements. Possible additional applications of the
128 assertion package format include management of embedded trust roots [XTAML] and
129 authorization policy information [XACML].

130 1.1 Namespaces

131 In this document, certain namespace prefixes represent specific XML namespaces.

132 All SAML protocol elements are defined using XML schema [XML-**Schema1**][XML-
133 **Schema2**]. For clarity unqualified elements in schema definitions are in the XML schema
134 namespace:

```
135     xmlns="http://www.w3.org/2001/XMLSchema"  
136     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

137 References to Security Assertion Markup Language schema defined herein use the prefix
138 `saml` : and are in the namespace:

```
139     xmlns:saml="http://www.oasis-open.org/committees/security/docs/draft-  
140     sstc-schema-assertion-19.xsd"
```

141 This namespace is also used for unqualified elements in message protocol examples.

142 The SAML schema specification uses some elements already defined in the XML
143 Signature namespace. The “XML Signature namespace” is represented by the prefix `ds`
144 and is declared as:

```
145     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
```

146 The “XML Signature schema” is defined in [XML-SIG-XSD] and the `<ds:KeyInfo>`
147 element (and all of its contents) are defined in [XML-SIG]§4.4.

148 The following schema defines the XML namespaces for the assertion schema:

```
149 <?xml version="1.0" encoding="UTF-8"?>  
150 <!-- edited with XML Spy v3.5 NT (http://www.xmlspy.com) by Phill Hallam-Baker  
151 (VeriSign Inc.) -->  
152 <schema  
153     targetNamespace="http://www.oasis-open.org/committees/security/docs/draft-  
154     sstc-schema-assertion-19.xsd"  
155     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"  
156     xmlns:saml="http://www.oasis-open.org/committees/security/docs/draft-sstc-  
157     schema-assertion-19.xsd"  
158     xmlns="http://www.w3.org/2001/XMLSchema"  
159     elementFormDefault="unqualified">  
160     <import namespace="http://www.w3.org/2000/09/xmldsig#"  
161         schemaLocation="xmldsig-core-schema.xsd"/>  
162     <annotation>  
163         <documentation>draft-sstc-schema-assertion-19.xsd</documentation>  
164     </annotation>
```

165 1.1.1 Basic Types

166 The types defined in this section define XML types that are considered part of the schema
167 as a whole rather than a component of a particular element. This allows for greater
168 consistency and avoids the need for extension schemas to redefine the same types.

169 **1.1.1.1 Simple Type IDType and Element <AssertionID>**

170 The IDType type is used for any data element that is an identifier for a specific data
171 object that is opaque to the SAML application. In the SAML specification the IDType
172 type is used declare and reference identifiers to assertions, requests and responses.

173 IDType identifiers MUST satisfy the following properties:

- 174 • Any party that assigns an IDType identifier to a data object MUST ensure that
175 there is negligible probability that that party or any other party will assign the
176 same identifier to a different data object.
- 177 • Where a data object specifies an IDType identifier that is to be an identifier for
178 that object there MUST be exactly one such declaration.

179 The mechanism by which the application ensures that the identifier is unique is left to the
180 implementation. In the case that a pseudorandom technique is employed the probability
181 of two randomly chosen identifiers being identical MUST be less than 2^{-128} and
182 SHOULD be less than 2^{-160} .

183 An IDType identifier MAY or MAY NOT be resolvable. In the case that the identifier is
184 resolvable (e.g. the identifier is a URL) the identifier MAY or MAY NOT resolve to the
185 data object identified.

186 The <AssertionID> element is used to specify an identifier that references a SAML
187 assertion.

188 The following schema specifies the <AssertionID> element and IDType type:

```
189 <element name="AssertionID" type="saml:IDType"/>  
190 <simpleType name="IDType">  
191 <restriction base="string"/>  
192 </simpleType>
```

193 **1.1.1.2 Simple Type DecisionType**

194 The DecisionType type reports the status of an authorization decision with respect to
195 a specific resource.

196 **Permit**

197 The specified action is permitted.

198 **Deny**

199 The specified action is denied.

200 **Indeterminate**

201 No statement is made as to whether the specified action is permitted or denied.

202 The following schema specifies the DecisionType type:

```
203 <simpleType name="DecisionType">  
204 <restriction base="string">
```

```
205     <enumeration value="Permit"/>
206     <enumeration value="Deny"/>
207     <enumeration value="Indeterminate"/>
208   </restriction>
209 </simpleType>
```

210 1.2 SAML Assertion

211 A SAML Assertion is specified by a single XML element whose type is derived from the
212 abstract XML type `AssertionAbstractType`. The abstract
213 `AssertionAbstractType` specifies the basic information that is common to all
214 SAML assertions. Instances of SAML assertions have concrete types that are extensions
215 of the base `AssertionAbstractType`.

216 1.2.1 Element `<Assertion>`

217 The element `<Assertion>` of abstract type `AssertionAbstractType` is used to
218 specify a SAML assertion:

219 `MajorVersion` [Required]

220 Each assertion MUST specify the SAML major version identifier. The identifier
221 for this version of SAML is 1.

222 `MinorVersion` [Required]

223 Each assertion MUST specify the SAML minor version identifier. The identifier
224 for this version of SAML is 0.

225 `AssertionID` [Required]

226 The `AssertionID` attribute specifies the assertion identifier.

227 `Issuer` [Required]

228 The `Issuer` attribute specifies the issuer of the assertion by means of a
229 `string[PHB1]`.

230 `IssueInstant` [Required]

231 The `IssueInstant` attribute specifies the time instant of issue in Universal
232 Coordinated Time (UTC).

233 `<Conditions>` [Optional]

234 The `<Conditions>` element specifies conditions that affect the validity of the
235 asserted statement.

236 `<Advice>` [Optional]

237 The `<Advice>` element specifies additional information related to the assertion
238 that may assist processing in certain situations but which can be ignored by
239 applications that do not support its use.

240 Processing of the `MajorVersion` and `MinorVersion` attributes is specified in
241 section 3 .

242 The following schema specifies the <Assertion> element and
243 AssertionAbstractType abstract type:

```
244 <element name="Assertion" type="saml:AssertionAbstractType"/>
245 <complexType name="AssertionAbstractType" abstract="true">
246   <sequence>
247     <element ref="saml:Conditions" minOccurs="0"/>
248     <element ref="saml:Advice" minOccurs="0"/>
249   </sequence>
250   <attribute name="MajorVersion" type="integer" use="required"/>
251   <attribute name="MinorVersion" type="integer" use="required"/>
252   <attribute name="AssertionID" type="saml:IDType" use="required"/>
253   <attribute name="Issuer" type="string" use="required"/>
254   <attribute name="IssueInstant" type="dateTime" use="required"/>
255 </complexType>
```

256 1.2.2 Element <SingleAssertion>

257 The element <SingleAssertion> of type SingleAssertionType is used to
258 specify a SAML assertion that contains a single statement that is one of the following:

259 <Statement>

260 A statement defined in an extension schema (see section 3.2)

261 <SubjectStatement>

262 A subject statement defined in an extension schema (see section 3.2)

263 <AuthenticationStatement>

264 An authentication statement

265 <AuthorizationStatement>

266 An authorization statement

267 <AttributeStatement>

268 An attribute statement.

269 The following schema specifies the <SingleAssertion> element:

```
270 <element name="SingleAssertion" type="saml:SingleAssertionType"/>
271 <complexType name="SingleAssertionType">
272   <complexContent>
273     <extension base="saml:AssertionAbstractType">
274       <choice>
275         <element ref="saml:Statement"/>
276         <element ref="saml:SubjectStatement"/>
277         <element ref="saml:AuthenticationStatement"/>
278         <element ref="saml:AuthorizationStatement"/>
279         <element ref="saml:AttributeStatement"/>
280       </choice>
281     </extension>
282   </complexContent>
283 </complexType>
```

284 1.2.3 Element <MultipleAssertion>

285 The element <MultipleAssertion> of type MultipleAssertionType is used
286 to specify a SAML assertion that contains any number of the statement elements
287 specified in section 1.2.2 above.

288 The following schema specifies the <MultipleAssertion> element:

```
289 <element name="MultipleAssertion" type="saml:MultipleAssertionType"/>
290 <complexType name="MultipleAssertionType">
291   <complexContent>
292     <extension base="saml:AssertionAbstractType">
293       <choice minOccurs="0" maxOccurs="unbounded">
294         <element ref="saml:Statement"/>
295         <element ref="saml:SubjectStatement"/>
296         <element ref="saml:AuthenticationStatement"/>
297         <element ref="saml:AuthorizationStatement"/>
298         <element ref="saml:AttributeStatement"/>
299       </choice>
300     </extension>
301   </complexContent>
302 </complexType>
```

303 1.2.4 Element <AssertionSpecifier>

304 The <AssertionSpecifier> element specifies an assertion either by reference or
305 by value. An assertion is specified by reference to the value of its AssertionID attribute.
306 An assertion is specified by value by including the entire assertion.

307 <AssertionID>
308 Specifies an assertion by reference to the value of its unique AssertionID
309 attribute.

310 <Assertion>
311 Specifies an assertion by value.

312 <SingleAssertion>
313 Specifies an assertion containing a single statement by value.

314 <MultipleAssertion>
315 Specifies an assertion containing multiple statements by value.

316 The following schema specifies the <AssertionSpecifier> element:

```
317 <element name="AssertionSpecifier" type="saml:AssertionSpecifierType"/>
318 <complexType name="AssertionSpecifierType">
319   <choice>
320     <element ref="saml:AssertionID"/>
321     <element ref="saml:Assertion"/>
322     <element ref="saml:SingleAssertion"/>
323     <element ref="saml:MultipleAssertion"/>
324   </choice>
325 </complexType>
```

326 1.2.5 Element <Statement>

327 The <Statement> element is an extension point that allows other assertion-based
328 applications to reuse the SAML assertion framework.

329 The following schema specifies the <Statement> element:

```
330 <element name="Statement" type="saml:StatementAbstractType"/>
331 <complexType name="StatementAbstractType" abstract="true"/>
```

332 **1.3 Subject Statement**

333 A subject statement is a statement that relates to the specific SAML subject specified by a
334 <Subject> element.

335 1.3.1 Element <SubjectStatement>

336 The <SubjectStatement> element is an extension point that allows other assertion-
337 based applications to reuse the SAML assertion framework.

338 The SubjectAssertionAbstractType type is an abstract type that extends the
339 AssertionAbstractType to include a <Subject> element.

340 The following schema defines the <SubjectAssertion> element and
341 SubjectAssertionAbstractType abstract type:

```
342 <element name="SubjectStatement" type="saml:SubjectStatementAbstractType"/>  
343 <complexType name="SubjectStatementAbstractType" abstract="true">  
344 <complexContent>  
345 <extension base="saml:StatementAbstractType">  
346 <sequence>  
347 <element ref="saml:Subject"/>  
348 </sequence>  
349 </extension>  
350 </complexContent>  
351 </complexType>
```

352 1.3.2 Element <Subject>

353 The <Subject> element specifies a party by any of the following means:

- 354 • A name.
- 355 • By information that allows the party to be authenticated.
- 356 • By reference to another assertion or by containment of another assertion.

357 If a <Subject> element contains more than one subject specification the issuer is
358 asserting that the statement is true for all of the subjects specified. For example if both a
359 <NameIdentifier> and a <SubjectConfirmation> element are present the
360 issuer is asserting that the statement is true of both parties.

361 The definition of a <Subject> element that intentionally identifies more than one
362 principal is deprecated.

363 The following schema defines the <Subject> element:

```
364 <element name="Subject" type="saml:SubjectType"/>  
365 <complexType name="SubjectType">  
366 <choice maxOccurs="unbounded">  
367 <element ref="saml:NameIdentifier"/>  
368 <element ref="saml:SubjectConfirmation"/>  
369 <element ref="saml:AssertionSpecifier"/>  
370 </choice>  
371 </complexType>
```

372 **1.3.2.1 Element <SubjectConfirmation>**

373 The <SubjectConfirmation> element specifies a subject by specifying data that
374 authenticates the subject.

375 <ConfirmationMethod> [Any number]

376 Each <AuthenticationMethod> element specifies a URI that identify a
377 protocol that may be used to authenticate the subject.

378 <SubjectConfirmationData> [Optional]

379 Each <SubjectConfirmationData> element specifies additional
380 authentication information used by a specific authentication protocol.

381 <ds:KeyInfo> [Optional]

382 An XML Signature <ds:KeyInfo> element that specifies a cryptographic key
383 held by the subject.

384 URIs identifying common authentication protocols are specified in Section 5 .

385 The following schema defines the <SubjectConfirmation> element:

```
386 <element name="SubjectConfirmation" type="saml:SubjectConfirmationType"/>  
387 <complexType name="SubjectConfirmationType">  
388 <sequence>  
389 <element ref="saml:ConfirmationMethod" maxOccurs="unbounded"/>  
390 <element name="SubjectConfirmationData" type="string" minOccurs="0"/>  
391 <element ref="ds:KeyInfo" minOccurs="0"/>  
392 </sequence>  
393 <!-- Need to modify this element-->  
394 </complexType>
```

395 **1.3.2.2 Element <NameIdentifier>**

396 The <NameIdentifier> element specifies a subject by a combination of a name and
397 a security domain.

398 The interpretation of the security domain and the name are left to individual
399 implementations, including issues of anonymity, pseudonymity, and the persistence of
400 the identifier with respect to the asserting and relying parties.

401 The following schema defines the <NameIdentifier> element:

```
402 <element name="NameIdentifier" type="saml:NameIdentifierType"/>  
403 <complexType name="NameIdentifierType">  
404 <attribute name="SecurityDomain" type="string"/>  
405 <attribute name="Name" type="string"/>  
406 </complexType>
```

407 **1.3.2.3 Element <ConfirmationMethod>**

408 The <ConfirmationMethod> element specifies the type of Authentication that took
409 place.

410 The following schema defines the <ConfirmationMethod> element:

```
411 <element name="ConfirmationMethod" type="anyURI"/>
```

412 **1.4 Authentication Assertion**

413 An authentication assertion is an assertion by the issuer that the subject was authenticated
414 by a particular means at a particular time.

415 **1.4.1 Element <AuthenticationStatement>**

416 The <AuthenticationStatement> element is of type
417 AuthenticationStatementType, which extends the
418 SubjectStatementAbstractType with the addition of the following elements:

419 AuthenticationMethod [Required]

420 The AuthenticationMethod attribute specifies the type of Authentication
421 that took place.

422 AuthenticationInstant [Required]

423 The AuthenticationInstant attribute specifies the time at which the
424 authentication took place.

425 <AuthenticationLocality> [Optional]

426 The <AuthenticationLocale> element specifies the DNS domain name
427 and IP address for the system entity that performed the authentication.

428 The following schema defines the <AuthenticationAssertion> assertion type:

```
429 <element name="AuthenticationStatement"  
430         type="saml:AuthenticationStatementType"/>  
431 <complexType name="AuthenticationStatementType">  
432   <complexContent>  
433     <extension base="saml:SubjectStatementAbstractType">  
434       <sequence>  
435         <element ref="saml:AuthenticationLocality" minOccurs="0"/>  
436       </sequence>  
437       <attribute name="AuthenticationMethod" type="anyURI"/>  
438       <attribute name="AuthenticationInstant" type="dateTime"/>  
439     </extension>  
440   </complexContent>  
441 </complexType>
```

442 **1.4.1.1 Element <AuthenticationLocality>**

443 The <AuthenticationLocality> element specifies the DNS domain name and IP
444 address for the system entity that was authenticated.

445 IPAddress [Optional]

446 The IPAddress attribute of the system entity that that was authenticated

447 DNSAddress [Required]

448 The DNSAddress attribute system entity that that was authenticated

449 *Note: This element is entirely advisory, since both these fields are quite easily “spoofed”*
450 *but current practice appears to require its inclusion.*

451 The following schema defines the <AuthenticationLocality> type:

```
452 <element name="AuthenticationLocality"  
453       type="saml:AuthenticationLocalityType"/>  
454 <complexType name="AuthenticationLocalityType">  
455   <attribute name="IPAddress" type="string" use="optional"/>  
456   <attribute name="DNSAddress" type="string" use="optional"/>  
457 </complexType>
```

458 1.5 Authorization Decision Statement

459 An authorization decision assertion contains a statement that assert that the request for
460 access by the specified subject to the specified object has resulted in the specified
461 decision on the basis of some optionally specified evidence.

462 1.5.1 Element <AuthorizationStatement>

463 The <AuthorizationStatement> element extends the
464 SubjectStatementAbstractType with the addition of the following elements:

465 **Resource** [Optional]

466 The Resource attribute specifies the resource by means of a URI.

467 **<Actions>** [Required]

468 The <Actions> element specifies the set of actions authorized for the specified
469 resource.

470 **Decision** [Optional]

471 The <Decision> attribute specifies the decision with respect to the specified
472 object.

473 **<Evidence>** [Any Number]

474 The <Evidence> element specifies a set of assertions that the issuer relied upon
475 in making the decision.

476 If the <Namespace> element is not specified the namespace specified in section 5.2.1 is
477 specified by default.

478 The following schema defines the <AuthorizationStatement> element:

```
479 <element name="AuthorizationStatement"  
480       type="saml:AuthorizationStatementType"/>  
481 <complexType name="AuthorizationStatementType">  
482   <complexContent>  
483     <extension base="saml:SubjectStatementAbstractType">  
484       <sequence>  
485         <element ref="saml:Actions"/>  
486         <element ref="saml:Evidence"  
487           minOccurs="0" maxOccurs="unbounded"/>  
488       </sequence>  
489       <attribute name="Resource" type="anyURI" use="optional"/>  
490       <attribute name="Decision"  
491         type="saml:DecisionType" use="optional"/>  
492     </extension>  
493   </complexContent>  
494 </complexType>
```

495 **1.5.1.1 Element <Actions>**

496 The `Actions` element specifies a set of actions within a specified action namespace.

497 **<Namespace>** [Optional]

498 The `<Namespace>` element specifies the namespace in which the specified
499 action elements are to be interpreted.

500 **<Action>** [One or more]

501 The `<Action>` element specifies the set of actions on the specified resource.

502 If the `Namespace` attribute is not specified the namespace specified in section 5.2.1 is
503 specified by default.

504 The following schema defines the `<Actions >` type:

```
505 <element name="Actions" type="saml:ActionsType"/>  
506 <complexType name="ActionsType">  
507 <sequence>  
508 <element ref="saml:Action" maxOccurs="unbounded"/>  
509 </sequence>  
510 <attribute name="Namespace" type="anyURI" use="optional"/>  
511 </complexType>  
512 <element name="Action" type="string"/>
```

513 **1.5.1.2 Element <Evidence>**

514 The `<Evidence>` element specifies an assertion that the issuer relied upon in issuing
515 the assertion in which the element is contained.

516 The statement of an assertion as evidence MAY affect the reliance agreement between
517 the client and service. For example in the case that the client presented an assertion to the
518 service in a request the service MAY use that assertion as evidence in making its
519 response without endorsing the assertion as valid either to the client or any third party.

520 The following schema defines the `<Evidence>` element:

```
521 <element name="Evidence" type="saml:AssertionSpecifierType"/>
```

522 **1.6 Attribute Statement**

523 An attribute assertion contains a statement that asserts that the specified subject is
524 associated with the specified attribute(s)

525 **1.6.1 Element <AttributeStatement>**

526 The `<AttributeStatement>` extends the `SubjectStatementAbstractType`
527 with the addition of the following element:

528 **<Attribute>** [One or More]

529 The `<Attribute>` element specifies an attribute of the assertion subject.

530 The following schema defines the `AttributeAssertionType` assertion type:

```
531 <element name="AttributeStatement" type="saml:AttributeStatementType"/>
532 <complexType name="AttributeStatementType">
533 <complexContent>
534 <extension base="saml:SubjectStatementAbstractType">
535 <sequence>
536 <element ref="saml:Attribute" maxOccurs="unbounded"/>
537 </sequence>
538 </extension>
539 </complexContent>
540 </complexType>
```

541 1.6.1.1 Element <AttributeDesignator>

542 The <AttributeDesignator> element specifies an attribute name within an
543 attribute namespace. The element is used in an attribute assertion query to request that
544 attribute values within a specific namespace be returned. The
545 <AttributeDesignator> element contains the following attributes:

546 **AttributeNameSpace** [Required]

547 The <AttributeNameSpace> attribute specifies the namespace in which the
548 <AttributeName> elements are interpreted.

549 **AttributeName** [Required]

550 The <AttributeName> attribute specifies the name of the attribute.

551 The following schema defines the <Attribute> element:

```
552 <element name="AttributeDesignator" type="saml:AttributeDesignatorType"/>
553 <complexType name="AttributeDesignatorType">
554 <attribute name="AttributeName" type="string"/>
555 <attribute name="AttributeNameSpace" type="anyURI"/>
556 </complexType>
```

557 1.6.1.2 Element <Attribute>

558 The <Attribute> element specifies an attribute of the assertion subject. The
559 <Attribute> element is an extension of the <AttributeDesignator> element
560 that allows the attribute value to be specified. The <Attribute> element contains the
561 following additional element:

562 **<AttributeValue>** [One or more]

563 Each <AttributeValue> element specifies a value of the attribute.

564 The following schema defines the <Attribute> element:

```
565 <element name="Attribute" type="saml:AttributeType"/>
566 <complexType name="AttributeType">
567 <complexContent>
568 <extension base="saml:AttributeDesignatorType">
569 <sequence>
570 <element ref="saml:AttributeValue"/>
571 </sequence>
572 </extension>
573 </complexContent>
574 </complexType>
```

575 **1.6.1.3 Type AttributeValueType**

576 An <AttributeValue> element is of type AttributeValueType. The
577 AttributeValueType allows the inclusion of any element in any namespace.

578 The following schema defines the AttributeValueType type:

```
579 <element name="AttributeValue" type="saml:AttributeValueType"/>  
580 <complexType name="AttributeValueType">  
581 <sequence>  
582 <any namespace="##any" processContents="lax"  
583 minOccurs="0" maxOccurs="unbounded"/>  
584 </sequence>  
585 </complexType>
```

586 **1.7 Conditions**

587 The validity of an assertion MAY be subject to a set of conditions. Each condition
588 evaluates to a value that is Valid, Invalid or Indeterminate. The validity status
589 of an assertion is the conjunction of the validity of each of the assertion conditions as
590 follows:

591 If any condition evaluates to Invalid.
592 The assertion status is Invalid

593 If no condition evaluates to Invalid and one or more conditions evaluate to
594 Indeterminate.
595 The assertion status is Indeterminate

596 If no conditions are specified or all the specified assertions evaluate to Valid.
597 The assertion status is Valid

598 **1.7.1 Element <Conditions>**

599 Assertion Conditions are contained in the <Conditions> element. SAML applications
600 MAY define additional elements using an extension schema. If an application encounters
601 an element contained within a <Conditions> element that is not understood the status
602 of the Condition MUST be considered Indeterminate.

603 <Condition> [Any Number]
604 The <Condition> element provides an extension point allowing extension
605 schemas to define new conditions.

606 <AudienceRestrictionCondition> [Any Number]
607 The <AudienceRestrictionCondition> condition specifies that the
608 assertion is addressed to a particular audience.

609 NotBefore [Optional]
610 The NotBefore attribute specifies the earliest time instant at which the
611 assertion is valid.

612 NotOnOrAfter [Optional]
613 The NotOnOrAfter attribute specifies the time instant at which the assertion
614 has expired.

615 The following schema defines the <Conditions> element:

```
616       <element name="Conditions" type="saml:ConditionsType"/>  
617       <complexType name="ConditionsType">  
618            <choice minOccurs="0" maxOccurs="unbounded">  
619                <element ref="saml:Condition"/>  
620                <element ref="saml:AudienceRestrictionCondition"/>  
621            </choice>  
622            <attribute name="NotBefore" type="dateTime" use="optional"/>  
623            <attribute name="NotOnOrAfter" type="dateTime" use="optional"/>  
624       </complexType>
```

625 1.7.1.1 Element <Condition>

626 The <Condition> element is of abstract type ConditionAbstractType and
627 serves as an extension point for new condition definitions.

628 The following schema defines the <Condition> element:

```
629       <element name="Condition" type="saml:ConditionAbstractType"/>  
630       <complexType name="ConditionAbstractType" abstract="true"/>
```

631 1.7.1.2 Attributes NotBefore and NotOnOrAfter

632 The NotBefore and NotOnOrAfter attributes specify limits on the validity of the
633 assertion.

634 The NotBefore attribute specifies the time instant at which the validity interval begins.
635 The NotOnOrAfter attribute specifies the time instant at which the validity interval
636 has ended

637 The NotBefore and NotOnOrAfter attributes are optional. If the value is either
638 omitted or equal to the start of the epoch it is unspecified. If the NotBefore attribute is
639 unspecified the assertion is valid at any time before the time instant specified by the
640 NotOnOrAfter attribute. If the NotOnOrAfter attribute is unspecified the assertion
641 is valid from the time instant specified by the NotBefore attribute with no expiry. If
642 neither attribute is specified the assertion is valid at any time.

643 In accordance with the XML Schemas Specification, all time instances are interpreted in
644 Universal Coordinated Time unless they explicitly indicate a time zone.

645 Implementations MUST NOT generate time instances that specify leap seconds.

646 1.7.2 Condition Type AudienceRestrictionConditionType

647 The <AudienceRestrictionCondition> element specifies that the assertion is
648 addressed to one or more specific audience(s). Although a party that is outside the
649 audience(s) specified is capable of drawing conclusions from an assertion, the issuer
650 explicitly makes no representation as to accuracy or trustworthiness to such a party.

651 An audience is identified by a URI namespace. The URI MAY identify a document that
652 describes the terms and conditions of audience membership.

653 The condition evaluates to True if and only if the relying party is a member of one or
654 more of the audiences specified.

655 The issuer of an assertion cannot prevent a party to whom it is disclosed making a
656 decision on the basis of the information provided. However, the
657 <AudienceRestrictionCondition> element allows the issuer to state explicitly
658 that no warranty is provided to such a party in a machine and human readable form.
659 While there can be no guarantee that a court would upholding such a warranty exclusion
660 in every circumstance, the probability of upholding the warranty exclusion is
661 considerably improved.

662 The following schema defines the <AudienceRestrictionCondition> condition
663 type:

```
664 <element name="AudienceRestrictionCondition"  
665         type="saml:AudienceRestrictionConditionType"/>  
666 <complexType name="AudienceRestrictionConditionType">  
667   <complexContent>  
668     <extension base="saml:ConditionAbstractType">  
669       <sequence>  
670         <element ref="saml:Audience"  
671             minOccurs="1" maxOccurs="unbounded"/>  
672       </sequence>  
673     </extension>  
674   </complexContent>  
675 </complexType>  
676 <element name="Audience" type="anyURI"/>
```

677 1.8 Advice

678 The Advice element is a general container for any additional information that does not
679 affect the semantics or validity of the assertion itself.

680 1.8.1 Element <Advice>

681 The <Advice> element permits additional information to be included in an assertion
682 that MAY be ignored by applications without affecting either the assertion semantics or
683 validity. Advice elements MAY be specified in an extension schema. The advice element
684 MAY be used to:

- 685 • Include evidence supporting the assertion claims to be cited, either directly
686 (through incorporating the claims) or indirectly (by reference to the supporting
687 assertions.
- 688 • State a proof of the assertion claims.
- 689 • Specify the timing and distribution points for updates to the assertion.

690 The following schema defines the <Advice> element:

```
691 <element name="Advice" type="saml:AdviceType"/>
```

```
692 <complexType name="AdviceType">
693   <sequence>
694     <choice minOccurs="0" maxOccurs="unbounded">
695       <element ref="saml:AssertionSpecifier"/>
696       <element ref="saml:AdviceElement"/>
697       <any namespace="##other" processContents="lax"/>
698     </choice>
699   </sequence>
700 </complexType>
701 <element name="AdviceElement" type="saml:AdviceAbstractType"/>
702 <complexType name="AdviceAbstractType"/>
703 </schema>
```

704 2 SAML Protocol

705 SAML Assertions may be generated and exchanged using a variety of protocols. The
706 bindings section of this document describes specific means of transporting SAML
707 assertions using existing widely deployed protocols.

708 SAML aware clients may in addition use the request protocol defined by the
709 <Request> and <Response> elements described in this section. A <Request>
710 from the client is followed by a <Response> from the service Figure 1.



711

712 Figure 1: SAML Request/Response Protocol

713 2.1 Namespaces

714 The namespaces of the protocol schema are the same as those of the assertion schema
715 defined in section 1.1 with the addition of the namespace for the protocol schema itself.

```
716 xmlns:samlp="http://www.oasis-
717 open.org/committees/security/docs/draft-sstc-schema-protocol-15.xsd"
```

718 The following schema defines the XML namespaces for the assertion schema:

```
719 <?xml version="1.0" encoding="UTF-8"?>
720 <!-- edited with XML Spy v3.5 NT (http://www.xmlspy.com) by Phill Hallam-Baker
721 (VeriSign Inc.) -->
722 <schema
723   targetNamespace="http://www.oasis-open.org/committees/security/docs/draft-
724   sstc-schema-protocol-19.xsd"
725   xmlns="http://www.w3.org/2001/XMLSchema"
726   xmlns:samlp="http://www.oasis-open.org/committees/security/docs/draft-sstc-
727   schema-protocol-19.xsd"
728   xmlns:saml="http://www.oasis-open.org/committees/security/docs/draft-sstc-
729   schema-assertion-19.xsd"
730   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
731   elementFormDefault="unqualified">
732   <import namespace="http://www.oasis-open.org/committees/security/docs/draft-
733   sstc-schema-assertion-19.xsd"
734     schemaLocation="draft-sstc-schema-assertion-19.xsd"/>
735   <import namespace="http://www.w3.org/2000/09/xmldsig#"
736     schemaLocation="xmldsig-core-schema.xsd"/>
737   <annotation>
738     <documentation>draft-sstc-schema-protocol-19.xsd</documentation>
739   </annotation>
```

740 2.1.1 Basic Types

741 The types defined in this section define XML types that are considered part of the schema
742 as a whole rather than a component of a particular element. This allows for greater
743 consistency and avoids the need for extension schemas to redefine the same types.

744 **2.1.1.1 Simple Type CompletenessSpecifierType**

745 The CompletenessSpecifierType type is used in a request to specify how a
746 service should respond in cases where a client makes a request and the client is not
747 authorized to receive part of the response. The CompletenessSpecifierType type
748 defines two possible values "Any" and "All".

749 **If Any is specified:**

750 The response contains the parts of the response that the client is authorized to
751 receive.

752 **If All is specified:**

753 The response is empty.

754 The following schema specifies the <CompletenessSpecifierType> type:

```
755 <simpleType name="CompletenessSpecifierType">  
756   <restriction base="string">  
757     <enumeration value="Partial"/>  
758     <enumeration value="AllOrNone"/>  
759   </restriction>  
760 </simpleType>
```

761 **2.1.1.2 Simple Type StatusCodeType**

762 The type StatusCodeType in a response specifies the status of the request. Four
763 status values are defined:

764 **Success**

765 The request succeeded.

766 **Failure**

767 The request could not be performed by the service.

768 **Error**

769 An error in the request prevented the service from processing it.

770 **Unknown**

771 The request failed for unknown reasons[PHB2]

772 The following schema specifies the <StatusCodeType> type:

```
773 <simpleType name="StatusCodeType">  
774   <restriction base="string">  
775     <enumeration value="Success"/>  
776     <enumeration value="Failure"/>  
777     <enumeration value="Error"/>  
778     <enumeration value="Unknown"/>  
779   </restriction>  
780 </simpleType>
```

781 **2.2 Request**

782 2.2.1 Abstract Type RequestAbstractType

783 All SAML requests are of types that are extensions of the RequestAbstractType
784 abstract type. The RequestAbstractType requires that all SAML requests specify
785 the version number of the SAML protocol and a request identifier:

786 RequestID [Required]

787 The RequestID attribute defines a unique identifier for the assertion request.
788 The RequestID element in a request MUST match the InResponseTo
789 element in the corresponding response.

790 MajorVersion [Required]

791 Each request MUST specify the SAML major version identifier. The identifier for
792 this version of SAML is 1.

793 MinorVersion [Required]

794 Each request MUST specify the SAML minor version identifier. The identifier for
795 this version of SAML is 0.

796 Processing of the MajorVersion and MinorVersion attributes is specified in
797 section 3 .

798 The following schema defines the RequestAbstractType abstract type:

```
799 <complexType name="RequestAbstractType" abstract="true">  
800 <attribute name="RequestID" type="saml:IDType" use="required"/>  
801 <attribute name="MajorVersion" type="integer" use="required"/>  
802 <attribute name="MinorVersion" type="integer" use="required"/>  
803 </complexType>
```

804 2.2.2 Element <Request>

805 The <Request> element specifies a SAML request. This may contain either a query or
806 a request for a specific assertion identified by AssertionID or
807 AssertionArtifact.

808 <Query>

809 The <Query> element is an extension point that allows extension schemas to
810 define new types of query.

811 <SubjectQuery>

812 The <SubjectQuery> element is an extension point that allows extension
813 schemas to define new types of query that specify a single SAML subject.

814 <AuthenticationQuery>

815 The <AuthenticationQuery> element specifies an authentication query.

816 <AttributeQuery>

817 The <AttributeQuery> element specifies an attribute query.

- 818 <AuthorizationQuery>
819 The <AuthorizationQuery> element specifies an authorization query.
- 820 <AssertionID>
821 The <AssertionID> element requests an assertion by its assertion identifier.
- 822 <AssertionArtifact>
823 The <AssertionArtifact> element requests an assertion by means of an
824 assertion artifact that relates to it.

825 The following schema defines the <Request> element:

```
826     <element name="Request" type="samlp:RequestType"/>  
827     <complexType name="RequestType">  
828       <complexContent>  
829         <extension base="samlp:RequestAbstractType">  
830           <choice>  
831             <element ref="samlp:Query"/>  
832             <element ref="samlp:SubjectQuery"/>  
833             <element ref="samlp:AuthenticationQuery"/>  
834             <element ref="samlp:AttributeQuery"/>  
835             <element ref="samlp:AuthorizationQuery"/>  
836             <element ref="saml:AssertionID" maxOccurs="unbounded"/>  
837             <element ref="samlp:AssertionArtifact" maxOccurs="unbounded"/>  
838           </choice>  
839         </extension>  
840       </complexContent>  
841     </complexType>  
842     <element name="AssertionArtifact" type="string"/>
```

843 2.2.3 Element <Query>

844 The <Query> element is an extension point that allows new SAML queries to be
845 defined. The Query <Query> element is of QueryAbstractType abstract type,
846 which is the base type from which all SAML request query elements are derived. The
847 QueryAbstractType abstract type contains no elements or attributes.

848 The following schema defines the <Query> element:

```
849     <element name="Query" type="samlp:QueryAbstractType"/>  
850     <complexType name="QueryAbstractType" abstract="true"/>
```

851 2.2.4 Element <SubjectQuery>

852 The <SubjectQuery> element is an extension point that allows new SAML queries
853 that specify a single SAML subject to be defined. The <SubjectQuery> element is of
854 SubjectQueryAbstractType type extends the Query type to specify a query with
855 a specific subject as its principal.

856 The following schema defines the <SubjectQuery> element:

```
857     <element name="SubjectQuery" type="samlp:SubjectQueryAbstractType"/>  
858     <complexType name="SubjectQueryAbstractType" abstract="true">  
859       <complexContent>  
860         <extension base="samlp:QueryAbstractType">  
861           <sequence>  
862             <element ref="saml:Subject"/>  
863           </sequence>  
864         </extension>  
865       </complexContent>
```

866 `</complexType>`

867 **2.3 Authentication Query**

868 The `<AuthenticationQuery>` element is used to make the query “What
869 authentication assertions are available for this Subject?”

870 The response will be in the form of an Authentication assertion.

871 **2.3.1 Element `<AuthenticationQuery>`**

872 An `<AuthenticationQuery>` contains all the elements and attributes of a
873 `<SubjectQuery>` and extends them as follows:

874 `<ConfirmationMethod>` [Optional]

875 The `<ConfirmationMethod>` element if present can be used to as a filter for
876 possible responses. This supports the query “What authentication assertions do
877 you have for this Subject with the following `ConfirmationMethod`?”

878 A SAML processor will return a certain number of Authentication Assertions in response
879 to an authentication query. The `<Subject>` of the returned assertions MUST be
880 identical to the `<Subject>` element of the Query. If the `<ConfirmationMethod>`
881 field is present in the Query at least one `<ConfirmationMethod>` field in the
882 response MUST match. There is no implication that all such assertions must be returned.

883 The following schema defines the `<AuthenticationQuery>` type:

```
884 <element name="AuthenticationQuery" type="saml:AuthenticationQueryType"/>  
885 <complexType name="AuthenticationQueryType">  
886 <complexContent>  
887 <extension base="saml:SubjectQueryAbstractType">  
888 <sequence>  
889 <element ref="saml:ConfirmationMethod" minOccurs="0"/>  
890 </sequence>  
891 </extension>  
892 </complexContent>  
893 </complexType>
```

894 **2.4 Attribute Query**

895 An Attribute Query is used to make the query “Return the requested attributes for this
896 Subject”

897 The response will be in the form of an Attribute Assertion.

898 **2.4.1 Element `<AttributeQuery>`**

899 An `<AttributeQuery>` element contains all the elements and attributes of a
900 `<SubjectQuery>` element and extends them as follows:

901 `<Attribute>` [Any number]

902 Each `<Attribute>` element specifies an attribute that is to be returned. If no
903 attributes are specified the scope of the query is implicit.

904 <CompletenessSpecifier> [Required]
905 The <CompletenessSpecifier> element specifies the desired behavior in
906 the case that access to some of the requested attributes is not authorized using the
907 CompletenessSpecifier.

908 The following schema defines the <AttributeQuery> element:

```
909 <element name="AttributeQuery" type="samlp:AttributeQueryType"/>  
910 <complexType name="AttributeQueryType">  
911 <complexContent>  
912 <extension base="samlp:SubjectQueryAbstractType">  
913 <sequence>  
914 <element ref="saml:AttributeDesignator"  
915 minOccurs="0" maxOccurs="unbounded"/>  
916 </sequence>  
917 <attribute name="CompletenessSpecifier"  
918 type="samlp:CompletenessSpecifierType" use="required"/>  
919 </extension>  
920 </complexContent>  
921 </complexType>
```

922 2.5 Authorization Query

923 An Authorization Query is used to make the query: "Should action(s) Y on resource Z be
924 allowed for subject S given evidence E?"

925 The answer comes in the form of an Authorization Decision assertion. The action(s) and
926 resource are optionally namespace-scoped..

927 2.5.1 Element <AuthorizationQuery>

928 An <AuthorizationQuery> element contains all the elements and attributes of a
929 <SubjectQuery> and extends them as follows:

930 <Object> [Required]
931 The <Object> element specifies the resource and action(s) for which
932 authorization is requested.

933 <Evidence> [Any number]
934 Each <Evidence> element specifies an assertion that the service may rely upon
935 in making its response.

936 The following schema defines the <AuthorizationQuery> type:

```
937 <element name="AuthorizationQuery" type="samlp:AuthorizationQueryType"/>  
938 <complexType name="AuthorizationQueryType">  
939 <complexContent>  
940 <extension base="samlp:SubjectQueryAbstractType">  
941 <sequence>  
942 <element ref="saml:Actions"/>  
943 <element ref="saml:Evidence"  
944 minOccurs="0" maxOccurs="unbounded"/>  
945 </sequence>  
946 <attribute name="Resource" type="anyURI"/>  
947 </extension>  
948 </complexContent>  
949 </complexType>
```

950 **2.6 Response**

951 2.6.1 Abstract Type ResponseAbstractType

952 The response to a Request is of a type that extends the ResponseAbstractType
953 abstract type. The ResponseAbstractType specifies information that is common to
954 all SAML responses, the SAML protocol version, the identifier of the response and the
955 identifier of the request that is responded to.

956 ResponseID [Required]

957 The ResponseID attribute defines a unique identifier for the assertion response.

958 InResponseTo [Required]

959 The InResponseTo attribute specifies the request to which the response
960 corresponds. The RequestID element MUST match the InResponseTo
961 element in the corresponding response.

962 MajorVersion [Required]

963 Each response MUST specify the SAML major version identifier. The identifier
964 for this version of SAML is 1.

965 MinorVersion [Required]

966 Each response MUST specify the SAML minor version identifier. The identifier
967 for this version of SAML is 0.

968 Processing of the MajorVersion and MinorVersion attributes is specified in
969 section 3 .

970 The following schema defines the ResponseAbstractType abstract type:

```
971 <complexType name="ResponseAbstractType" abstract="true">  
972   <attribute name="ResponseID" type="saml:IDType" use="required"/>  
973   <attribute name="InResponseTo" type="saml:IDType" use="required"/>  
974   <attribute name="MajorVersion" type="integer" use="required"/>  
975   <attribute name="MinorVersion" type="integer" use="required"/>  
976 </complexType>
```

977 2.6.2 Element <Response>

978 The <Response> element extends the ResponseAbstractType and specifies
979 the status of the corresponding SAML Request and a list of zero or more assertions that
980 answer the request.

981 The following schema defines the <Response> element:

```
982 <element name="Response" type="samlp:ResponseType"/>  
983 <complexType name="ResponseType">  
984   <complexContent>  
985     <extension base="samlp:ResponseAbstractType">  
986       <choice minOccurs="0" maxOccurs="unbounded">  
987         <element ref="saml:Assertion"/>  
988         <element ref="saml:SingleAssertion"/>  
989         <element ref="saml:MultipleAssertion"/>  
990       </choice>
```

```
991         <attribute name="StatusCode"  
992             type="samlp:StatusCodeType" use="required"/>  
993     </extension>  
994 </complexContent>  
995 </complexType>  
996 </schema>
```

997 3 SAML Versions and Extension Schemas

998 3.1 SAML Version

999 Version Numbers are defined in the following elements:

- 1000 • Assertion
- 1001 • Request
- 1002 • Response

1003 The version number of the SAML assertion is independent of the version number of the
1004 SAML protocol.

1005 Each Version number consists of a major version number and a minor version number,
1006 both of which are integers. In accordance with industry practice a version number
1007 SHOULD be presented to the user in the form *Major.Minor*. This document describes
1008 version number 1.0 of the SAML Assertion and protocol.

1009 The version number *Major_B.Minor_B* is higher than the version number *Major_A.Minor_A* if
1010 and only if:

$$1011 \quad Major_B > Major_A \vee ((Major_B = Major_A) \wedge Minor_B = Minor_A)$$

1012 New versions of SAML SHALL assign version numbers to assertions, requests and
1013 responses that are the same as or higher than the corresponding version number in the
1014 SAML version that immediately preceded it.

1015 New versions of SAML SHALL assign new version numbers as follows:

1016 **Documentation change:** $(Major_B = Major_A) \wedge (Minor_B = Minor_A)$
1017 If the major and minor version numbers are unchanged, the new version *B* only
1018 introduces changes to the documentation that raise no compatibility issues with an
1019 implementation of version *A*.

1020 **Minor upgrade:** $(Major_B = Major_A) \wedge (Minor_B > Minor_A)$
1021 If the major version number of versions *A* and *B* are the same and the minor
1022 version number of *A* is higher than that of *B* the new SAML version may
1023 introduce changes to the SAML schema and semantics but any changes that are
1024 introduced in *B* SHALL be compatible with version *A*.

1025 **Major upgrade:** $Major_B > Major_A$
1026 If the major version of *B* number is higher than the major version of *A*, Version *B*
1027 may introduce changes to the SAML schema and semantics that are incompatible
1028 with *A*.

1029 **3.1.1 Assertion Version**

1030 A SAML application **MUST NOT** issue any assertion whose version number is not
1031 supported.

1032 A SAML application **MUST** reject any assertion whose major version number is not
1033 supported.

1034 A SAML application **MAY** reject any assertion whose version number is higher than the
1035 highest supported version.

1036 **3.1.2 Request Version**

1037 A SAML application **SHOULD** issue requests that specify the highest SAML version
1038 supported by both the sender and recipient.

1039 If the SAML application does not know the capabilities of the recipient it should assume
1040 that it supports the highest SAML version supported by the sender.

1041 **3.1.3 Response Version**

1042 A SAML application **MUST NOT** issue responses that specify a higher SAML version
1043 number than the corresponding request.

1044 A SAML application **MUST NOT** issue a response that has a major version number that
1045 is lower than the major version number of the corresponding request except to report the
1046 error `RequestVersionTooHigh`.

1047 Incompatible protocol versions may cause the following errors to be reported:

1048 `RequestVersionTooHigh`

1049 The protocol version specified in the request is a major upgrade from the highest
1050 protocol version supported by the responder.

1051 `RequestVersionTooLow`

1052 The responder cannot respond to the particular request using the SAML version
1053 specified in the request because it is too low.

1054 `RequestVersionDepricated`

1055 The responder does not respond to any requests with the protocol version
1056 specified in the request.

1057 **3.2 Schema Extension**

1058 The SAML schema is designed to support extensibility.

1059 3.2.1 Use of sub-typing and substitution groups

1060 XML provides two principle mechanisms for specifying an element of an extended type,
1061 sub-typing and substitution groups. For example an element of the new assertion type
1062 `NewStatementType` defined in the schema `new`: may be specified using the
1063 `xsi:type` attribute as follows:

```
1064 <Assertion xmlns="...SAML.xsd" xmlns:new="...New.xsd" ...>  
1065   <Statement xsi:type="new:NewStatementType">  
1066     ...  
1067   </Statement>  
1068 </Assertion>
```

1069 Alternatively the new schema may define a `<NewStatment>` element of
1070 `NewStatementType` that is a member of the `Statement` substitution group:

```
1071 <element "NewStatement" type="new:NewStatementType"  
1072   substitutionGroup="saml:Statement"/>
```

1073 The substitution group declaration allows the `<NewStatement>` element to be used
1074 anywhere the SAML `<Statement>` element may be used:

```
1075 <Assertion xmlns="...SAML.xsd" xmlns:new="...New.xsd" ...>  
1076   <new:NewStatement>  
1077     ...  
1078   </new:NewStatement>  
1079 </Assertion>
```

1080 The choice of using sub-typing or substitution groups has no effect on the semantics of
1081 the XML document but does have implications for interoperability:

1082 **Advantages of using sub-typing:**

- 1083 • A document can be partially interpreted by a parser that does not have access to
1084 the extension schema.
- 1085 • At the time of writing some XML validating parsers do not support substitution
1086 groups whereas the `xsi:type` attribute is widely supported.

1087 **Advantages of using substitution groups:**

- 1088 • A document can be explained without the need to explain the importance of the
1089 `xsi:type` attribute.

1090 The SAML schema is designed to permit processing of the assertion package to be
1091 separated from the processing of the statements it contains if either extension mechanism
1092 is used.

1093 3.2.2 Extending the SAML Assertion Schema

1094 The following elements are intended for use as extension points in an extension schema.

Element	Purpose
<code><Assertion></code>	Define new assertion package

<Statement>	Define new assertion statement
<SubjectStatement>	Define new assertion statement that takes a single subject as the subject.
<AttributeValue>	Define a structured attribute value
<Condition>	Define a new condition
<AdviceElement>	Define a new advice element

1095 In addition the following elements that define specific instances of an assertion element
1096 may be extended to add specific additional functionality:

- 1097 • SingleAssertion
- 1098 • MultipleAssertion
- 1099 • AuthenticationStatement
- 1100 • AuthorizationStatement
- 1101 • AttributeStatement
- 1102 • AudienceRestrictionCondition

1103 3.2.3 Extending the SAML Protocol Schema

1104 The following elements are intended for use as extension points in an extension schema.

Element	Purpose
<Request>	Define new request type
<Query>	Define new query type
<SubjectQuery>	Define new subject query type
<Response>	Define a new response type

1105 In addition the following elements that define specific instances of an assertion element
1106 may be extended to add specific additional functionality:

- 1107 • <AuthenticationQuery>
- 1108 • <AuthorizationQuery>
- 1109 • <AttributeQuery>

1110 **4 References**

- 1111 **[Kerberos]** Needham, R., and M. Schroeder, *Using Encryption for*
 1112 *Authentication in Large Networks of Computers*, Communications
 1113 of the ACM, Vol. 21 (12), pp. 993-999, December 1978.
- 1114 **[SAML-USE]** *TBS*
- 1115 **[PKCS1]** Kaliski, B., *PKCS #1: RSA Encryption Version 2.0*, RSA
 1116 Laboratories, also IETF RFC 2437, October 1998.
- 1117 **[RFC-2104]** Krawczyk, H., Bellare, M. and R. Canetti, *HMAC: Keyed Hashing*
 1118 *for Message Authentication*, IETF RFC 2104, February 1997.
- 1119 **[SOAP]** D. Box, D Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn,
 1120 H. Frystyk Nielsen, S Thatte, D. Winer. *Simple Object Access*
 1121 *Protocol (SOAP) 1.1*, W3C Note 08 May 2000,
 1122 <http://www.w3.org/TR/SOAP>
- 1123 **[WSSL]** E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, *Web*
 1124 *Services Description Language (WSDL) 1.0* September 25, 2000,
 1125 <http://msdn.microsoft.com/xml/general/wSDL.asp>
- 1126 **[XACML]** *TBS*
- 1127 **[XTAML]** P. Hallam-Baker, *XML Trust Axiom Markup Language 1.0*,
 1128 VeriSign Inc. September 2001. <http://www.xmltrustcenter.org/>
- 1129 **[XML-SIG]** D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon.
 1130 *XML-Signature Syntax and Processing*, World Wide Web
 1131 Consortium. <http://www.w3.org/TR/xmlsig-core/>
- 1132 **[XML-SIG-XSD]** XML Signature Schema available from
 1133 [http://www.w3.org/TR/2000/CR-xmlsig-core-20001031/xmlsig-](http://www.w3.org/TR/2000/CR-xmlsig-core-20001031/xmlsig-core-schema.xsd)
 1134 [core-schema.xsd](http://www.w3.org/TR/2000/CR-xmlsig-core-20001031/xmlsig-core-schema.xsd).
- 1135 **[XML-Enc]** *XML Encryption Specification*, In development.
- 1136 **[XML-Schema1]** H. S. Thompson, D. Beech, M. Maloney, N. Mendelsohn. *XML*
 1137 *Schema Part 1: Structures*, W3C Working Draft 22 September
 1138 2000, <http://www.w3.org/TR/2000/WD-xmlschema-1-20000922/>,
 1139 latest draft at <http://www.w3.org/TR/xmlschema-1/>
- 1140 **[XML-Schema2]** P. V. Biron, A. Malhotra, *XML Schema Part 2: Datatypes*; W3C
 1141 Working Draft 22 September 2000,
 1142 <http://www.w3.org/TR/2000/WD-xmlschema-2-20000922/>, latest
 1143 draft at <http://www.w3.org/TR/xmlschema-2/>

1144 **5 Identifiers**

1145 **5.1 Authentication Protocol Identifiers**

1146 5.1.1 SAML Artifact

1147 5.1.2 Assertion Bearer

1148 5.1.3 User Name and Password (Pass-through)

1149 5.1.4 User Name and Password (One-Way-Function SHA-1)

1150 5.1.5 Kerberos

1151 5.1.6 SSL/TLS Certificate Based Client Authentication

1152 5.1.7 Object Authenticator (SHA-1)

1153 **5.2 Action Identifiers**

1154 5.2.1 Read/Write/Execute/Delete/Control

1155 5.2.2 Read/Write/Execute/Delete/Control with Negation

1156 5.2.3 Get/Head/Put/Post

1157 5.2.4 UNIX file Permissions

1159 **6.1 Assertion Schema**

```

1160 <?xml version="1.0" encoding="UTF-8"?>
1161 <!-- edited with XML Spy v3.5 NT (http://www.xmlspy.com) by Phill Hallam-Baker
1162 (VeriSign Inc.) -->
1163 <schema
1164     targetNamespace="http://www.oasis-open.org/committees/security/docs/draft-
1165 sstc-schema-assertion-19.xsd"
1166     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
1167     xmlns:saml="http://www.oasis-open.org/committees/security/docs/draft-sstc-
1168 schema-assertion-19.xsd"
1169     xmlns="http://www.w3.org/2001/XMLSchema"
1170     elementFormDefault="unqualified">
1171     <import namespace="http://www.w3.org/2000/09/xmldsig#"
1172         schemaLocation="xmldsig-core-schema.xsd"/>
1173     <annotation>
1174         <documentation>draft-sstc-schema-assertion-19.xsd</documentation>
1175     </annotation>
1176     <element name="AssertionID" type="saml:IDType"/>
1177     <simpleType name="IDType">
1178         <restriction base="string"/>
1179     </simpleType>
1180     <simpleType name="DecisionType">
1181         <restriction base="string">
1182             <enumeration value="Permit"/>
1183             <enumeration value="Deny"/>
1184             <enumeration value="Indeterminate"/>
1185         </restriction>
1186     </simpleType>
1187     <element name="Assertion" type="saml:AssertionAbstractType"/>
1188     <complexType name="AssertionAbstractType" abstract="true">
1189         <sequence>
1190             <element ref="saml:Conditions" minOccurs="0"/>
1191             <element ref="saml:Advice" minOccurs="0"/>
1192         </sequence>
1193         <attribute name="MajorVersion" type="integer" use="required"/>
1194         <attribute name="MinorVersion" type="integer" use="required"/>
1195         <attribute name="AssertionID" type="saml:IDType" use="required"/>
1196         <attribute name="Issuer" type="string" use="required"/>
1197         <attribute name="IssueInstant" type="dateTime" use="required"/>
1198     </complexType>
1199     <element name="SingleAssertion" type="saml:SingleAssertionType"/>
1200     <complexType name="SingleAssertionType">
1201         <complexContent>
1202             <extension base="saml:AssertionAbstractType">
1203                 <choice>
1204                     <element ref="saml:Statement"/>
1205                     <element ref="saml:SubjectStatement"/>
1206                     <element ref="saml:AuthenticationStatement"/>
1207                     <element ref="saml:AuthorizationStatement"/>
1208                     <element ref="saml:AttributeStatement"/>
1209                 </choice>
1210             </extension>
1211         </complexContent>
1212     </complexType>
1213     <element name="MultipleAssertion" type="saml:MultipleAssertionType"/>
1214     <complexType name="MultipleAssertionType">
1215         <complexContent>
1216             <extension base="saml:AssertionAbstractType">
1217                 <choice minOccurs="0" maxOccurs="unbounded">
1218                     <element ref="saml:Statement"/>
1219                     <element ref="saml:SubjectStatement"/>
1220                     <element ref="saml:AuthenticationStatement"/>
1221                     <element ref="saml:AuthorizationStatement"/>
1222                     <element ref="saml:AttributeStatement"/>
1223                 </choice>

```

```

1224         </extension>
1225     </complexContent>
1226 </complexType>
1227 <element name="AssertionSpecifier" type="saml:AssertionSpecifierType"/>
1228 <complexType name="AssertionSpecifierType">
1229     <choice>
1230         <element ref="saml:AssertionID"/>
1231         <element ref="saml:Assertion"/>
1232         <element ref="saml:SingleAssertion"/>
1233         <element ref="saml:MultipleAssertion"/>
1234     </choice>
1235 </complexType>
1236 <element name="Statement" type="saml:StatementAbstractType"/>
1237 <complexType name="StatementAbstractType" abstract="true"/>
1238 <element name="SubjectStatement" type="saml:SubjectStatementAbstractType"/>
1239 <complexType name="SubjectStatementAbstractType" abstract="true">
1240     <complexContent>
1241         <extension base="saml:StatementAbstractType">
1242             <sequence>
1243                 <element ref="saml:Subject"/>
1244             </sequence>
1245         </extension>
1246     </complexContent>
1247 </complexType>
1248 <element name="Subject" type="saml:SubjectType"/>
1249 <complexType name="SubjectType">
1250     <choice maxOccurs="unbounded">
1251         <element ref="saml:NameIdentifier"/>
1252         <element ref="saml:SubjectConfirmation"/>
1253         <element ref="saml:AssertionSpecifier"/>
1254     </choice>
1255 </complexType>
1256 <element name="SubjectConfirmation" type="saml:SubjectConfirmationType"/>
1257 <complexType name="SubjectConfirmationType">
1258     <sequence>
1259         <element ref="saml:ConfirmationMethod" maxOccurs="unbounded"/>
1260         <element name="SubjectConfirmationData" type="string" minOccurs="0"/>
1261         <element ref="ds:KeyInfo" minOccurs="0"/>
1262     </sequence>
1263     <!-- Need to modify this element-->
1264 </complexType>
1265 <element name="NameIdentifier" type="saml:NameIdentifierType"/>
1266 <complexType name="NameIdentifierType">
1267     <attribute name="SecurityDomain" type="string"/>
1268     <attribute name="Name" type="string"/>
1269 </complexType>
1270 <element name="ConfirmationMethod" type="anyURI"/>
1271 <element name="AuthenticationStatement"
1272     type="saml:AuthenticationStatementType"/>
1273 <complexType name="AuthenticationStatementType">
1274     <complexContent>
1275         <extension base="saml:SubjectStatementAbstractType">
1276             <sequence>
1277                 <element ref="saml:AuthenticationLocality" minOccurs="0"/>
1278             </sequence>
1279             <attribute name="AuthenticationMethod" type="anyURI"/>
1280             <attribute name="AuthenticationInstant" type="dateTime"/>
1281         </extension>
1282     </complexContent>
1283 </complexType>
1284 <element name="AuthenticationLocality"
1285     type="saml:AuthenticationLocalityType"/>
1286 <complexType name="AuthenticationLocalityType">
1287     <attribute name="IPAddress" type="string" use="optional"/>
1288     <attribute name="DNSAddress" type="string" use="optional"/>
1289 </complexType>
1290 <element name="AuthorizationStatement"
1291     type="saml:AuthorizationStatementType"/>
1292 <complexType name="AuthorizationStatementType">
1293     <complexContent>
1294         <extension base="saml:SubjectStatementAbstractType">

```

```

1295         <sequence>
1296             <element ref="saml:Actions"/>
1297             <element ref="saml:Evidence"
1298                 minOccurs="0" maxOccurs="unbounded"/>
1299         </sequence>
1300         <attribute name="Resource" type="anyURI" use="optional"/>
1301         <attribute name="Decision"
1302             type="saml:DecisionType" use="optional"/>
1303     </extension>
1304 </complexContent>
1305 </complexType>
1306 <element name="Actions" type="saml:ActionsType"/>
1307 <complexType name="ActionsType">
1308     <sequence>
1309         <element ref="saml:Action" maxOccurs="unbounded"/>
1310     </sequence>
1311     <attribute name="Namespace" type="anyURI" use="optional"/>
1312 </complexType>
1313 <element name="Action" type="string"/>
1314 <element name="Evidence" type="saml:AssertionSpecifierType"/>
1315 <element name="AttributeStatement" type="saml:AttributeStatementType"/>
1316 <complexType name="AttributeStatementType">
1317     <complexContent>
1318         <extension base="saml:SubjectStatementAbstractType">
1319             <sequence>
1320                 <element ref="saml:Attribute" maxOccurs="unbounded"/>
1321             </sequence>
1322         </extension>
1323     </complexContent>
1324 </complexType>
1325 <element name="AttributeDesignator" type="saml:AttributeDesignatorType"/>
1326 <complexType name="AttributeDesignatorType">
1327     <attribute name="AttributeName" type="string"/>
1328     <attribute name="AttributeNamespace" type="anyURI"/>
1329 </complexType>
1330 <element name="Attribute" type="saml:AttributeType"/>
1331 <complexType name="AttributeType">
1332     <complexContent>
1333         <extension base="saml:AttributeDesignatorType">
1334             <sequence>
1335                 <element ref="saml:AttributeValue"/>
1336             </sequence>
1337         </extension>
1338     </complexContent>
1339 </complexType>
1340 <element name="AttributeValue" type="saml:AttributeValueType"/>
1341 <complexType name="AttributeValueType">
1342     <sequence>
1343         <any namespace="##any" processContents="lax"
1344             minOccurs="0" maxOccurs="unbounded"/>
1345     </sequence>
1346 </complexType>
1347 <element name="Conditions" type="saml:ConditionsType"/>
1348 <complexType name="ConditionsType">
1349     <choice minOccurs="0" maxOccurs="unbounded">
1350         <element ref="saml:Condition"/>
1351         <element ref="saml:AudienceRestrictionCondition"/>
1352     </choice>
1353     <attribute name="NotBefore" type="dateTime" use="optional"/>
1354     <attribute name="NotOnOrAfter" type="dateTime" use="optional"/>
1355 </complexType>
1356 <element name="Condition" type="saml:ConditionAbstractType"/>
1357 <complexType name="ConditionAbstractType" abstract="true"/>
1358 <element name="AudienceRestrictionCondition"
1359     type="saml:AudienceRestrictionConditionType"/>
1360 <complexType name="AudienceRestrictionConditionType">
1361     <complexContent>
1362         <extension base="saml:ConditionAbstractType">
1363             <sequence>
1364                 <element ref="saml:Audience"
1365                     minOccurs="1" maxOccurs="unbounded"/>

```

```

1366         </sequence>
1367     </extension>
1368 </complexContent>
1369 </complexType>
1370 <element name="Audience" type="anyURI"/>
1371 <element name="Advice" type="saml:AdviceType"/>
1372 <complexType name="AdviceType">
1373     <sequence>
1374         <choice minOccurs="0" maxOccurs="unbounded">
1375             <element ref="saml:AssertionSpecifier"/>
1376             <element ref="saml:AdviceElement"/>
1377             <any namespace="##other" processContents="lax"/>
1378         </choice>
1379     </sequence>
1380 </complexType>
1381 <element name="AdviceElement" type="saml:AdviceAbstractType"/>
1382 <complexType name="AdviceAbstractType"/>
1383 </schema>

```

1384 6.2 Protocol Schema

```

1385 <?xml version="1.0" encoding="UTF-8"?>
1386 <!-- edited with XML Spy v3.5 NT (http://www.xmlspy.com) by Phill Hallam-Baker
1387 (VeriSign Inc.) -->
1388 <schema
1389     targetNamespace="http://www.oasis-open.org/committees/security/docs/draft-
1390 sstc-schema-protocol-19.xsd"
1391     xmlns="http://www.w3.org/2001/XMLSchema"
1392     xmlns:samlp="http://www.oasis-open.org/committees/security/docs/draft-sstc-
1393 schema-protocol-19.xsd"
1394     xmlns:saml="http://www.oasis-open.org/committees/security/docs/draft-sstc-
1395 schema-assertion-19.xsd"
1396     xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
1397     elementFormDefault="unqualified">
1398     <import namespace="http://www.oasis-open.org/committees/security/docs/draft-
1399 sstc-schema-assertion-19.xsd"
1400         schemaLocation="draft-sstc-schema-assertion-19.xsd"/>
1401     <import namespace="http://www.w3.org/2000/09/xmldsig#"
1402         schemaLocation="xmldsig-core-schema.xsd"/>
1403     <annotation>
1404         <documentation>draft-sstc-schema-protocol-19.xsd</documentation>
1405     </annotation>
1406     <simpleType name="CompletenessSpecifierType">
1407         <restriction base="string">
1408             <enumeration value="Partial"/>
1409             <enumeration value="AllOrNone"/>
1410         </restriction>
1411     </simpleType>
1412     <simpleType name="StatusCodeType">
1413         <restriction base="string">
1414             <enumeration value="Success"/>
1415             <enumeration value="Failure"/>
1416             <enumeration value="Error"/>
1417             <enumeration value="Unknown"/>
1418         </restriction>
1419     </simpleType>
1420     <complexType name="RequestAbstractType" abstract="true">
1421         <attribute name="RequestID" type="saml:IDType" use="required"/>
1422         <attribute name="MajorVersion" type="integer" use="required"/>
1423         <attribute name="MinorVersion" type="integer" use="required"/>
1424     </complexType>
1425     <element name="Request" type="samlp:RequestType"/>
1426     <complexType name="RequestType">
1427         <complexContent>
1428             <extension base="samlp:RequestAbstractType">
1429                 <choice>
1430                     <element ref="samlp:Query"/>
1431                     <element ref="samlp:SubjectQuery"/>
1432                     <element ref="samlp:AuthenticationQuery"/>
1433                     <element ref="samlp:AttributeQuery"/>
1434                     <element ref="samlp:AuthorizationQuery"/>

```

```

1435         <element ref="saml:AssertionID" maxOccurs="unbounded"/>
1436         <element ref="sampl:AssertionArtifact" maxOccurs="unbounded"/>
1437     </choice>
1438 </extension>
1439 </complexContent>
1440 </complexType>
1441 <element name="AssertionArtifact" type="string"/>
1442 <element name="Query" type="sampl:QueryAbstractType"/>
1443 <complexType name="QueryAbstractType" abstract="true"/>
1444 <element name="SubjectQuery" type="sampl:SubjectQueryAbstractType"/>
1445 <complexType name="SubjectQueryAbstractType" abstract="true">
1446     <complexContent>
1447         <extension base="sampl:QueryAbstractType">
1448             <sequence>
1449                 <element ref="saml:Subject"/>
1450             </sequence>
1451         </extension>
1452     </complexContent>
1453 </complexType>
1454 <element name="AuthenticationQuery" type="sampl:AuthenticationQueryType"/>
1455 <complexType name="AuthenticationQueryType">
1456     <complexContent>
1457         <extension base="sampl:SubjectQueryAbstractType">
1458             <sequence>
1459                 <element ref="saml:ConfirmationMethod" minOccurs="0"/>
1460             </sequence>
1461         </extension>
1462     </complexContent>
1463 </complexType>
1464 <element name="AttributeQuery" type="sampl:AttributeQueryType"/>
1465 <complexType name="AttributeQueryType">
1466     <complexContent>
1467         <extension base="sampl:SubjectQueryAbstractType">
1468             <sequence>
1469                 <element ref="saml:AttributeDesignator"
1470                     minOccurs="0" maxOccurs="unbounded"/>
1471             </sequence>
1472             <attribute name="CompletenessSpecifier"
1473                 type="sampl:CompletenessSpecifierType" use="required"/>
1474         </extension>
1475     </complexContent>
1476 </complexType>
1477 <element name="AuthorizationQuery" type="sampl:AuthorizationQueryType"/>
1478 <complexType name="AuthorizationQueryType">
1479     <complexContent>
1480         <extension base="sampl:SubjectQueryAbstractType">
1481             <sequence>
1482                 <element ref="saml:Actions"/>
1483                 <element ref="saml:Evidence"
1484                     minOccurs="0" maxOccurs="unbounded"/>
1485             </sequence>
1486             <attribute name="Resource" type="anyURI"/>
1487         </extension>
1488     </complexContent>
1489 </complexType>
1490 <complexType name="ResponseAbstractType" abstract="true">
1491     <attribute name="ResponseID" type="saml:IDType" use="required"/>
1492     <attribute name="InResponseTo" type="saml:IDType" use="required"/>
1493     <attribute name="MajorVersion" type="integer" use="required"/>
1494     <attribute name="MinorVersion" type="integer" use="required"/>
1495 </complexType>
1496 <element name="Response" type="sampl:ResponseType"/>
1497 <complexType name="ResponseType">
1498     <complexContent>
1499         <extension base="sampl:ResponseAbstractType">
1500             <choice minOccurs="0" maxOccurs="unbounded">
1501                 <element ref="saml:Assertion"/>
1502                 <element ref="saml:SingleAssertion"/>
1503                 <element ref="saml:MultipleAssertion"/>
1504             </choice>
1505             <attribute name="StatusCode"

```

```
1506         type="sampl:StatusCodeType" use="required"/>
1507     </extension>
1508 </complexContent>
1509 </complexType>
1510 </schema>
```

Page: 7

[PHB1] Need some better text here n'est pas?

Page: 21

[PHB2]Need to have text for these, how exactly does failure differ from error?