



Conformance Program Specification for the OASIS Security Assertion Markup Language (SAML) V1.1

Committee Specification, 27 May 2003

Document identifier:

sstc-saml-conform-1.1-cs-01

Location:

http://www.oasis-open.org/committees/documents.php?wg_abbrev=security

Editors:

Eve Maler, Sun Microsystems (eve.maler@sun.com)
Prateek Mishra, Netegrity (pmishra@netegrity.com)
Robert Philpott, RSA Security (rphilpott@rsasecurity.com)

Contributors:

Irving Reid, Baltimore Technologies
Krishna Sankar, Cisco Systems
Hal Lockhart, BEA Systems (formerly of Entegriy Solutions)
Robert Griffin, Entrust (former editor)
Marc Chanliau, Netegrity
Lynne Rosenthal, NIST
Mark Skall, NIST
Darren Platt, formerly with RSA Security
Charles Norwood, SAIC
Emily Xu, Sun Microsystems
Sai Allarvarpu, Sun Microsystems
Mike Myers, Traceroute Security
Mark O'Neill, Vordel
Tony Palmer, Vordel

Abstract:

This specification describes the program and technical requirements for SAML conformance.

Status:

This document is a **Committee Specification** of the OASIS Security Services Technical Committee. This document is updated periodically on no particular schedule. Send comments to the editors.

Committee members should send comments on this specification to the security-services@lists.oasis-open.org list. Others should subscribe to and send comments to the security-services-comment@lists.oasis-open.org list. To subscribe, send an email message to

38 security-services-comment-request@lists.oasis-open.org with the word "subscribe" as the body of
39 the message.

40 For information on whether any patents have been disclosed that may be essential to
41 implementing this specification, and any offers of patent licensing terms, please refer to the
42 Intellectual Property Rights section of the Security Services TC web page ([http://www.oasis-
open.org/committees/security/](http://www.oasis-
43 open.org/committees/security/)).

44 For information on errata discovered in this specification, please refer to the most recent errata
45 document which can be found in the document repository at the Security Services TC web page
46 (<http://www.oasis-open.org/committees/security/>).

Table of Contents

48	1	Introduction.....	5
49	1.1	Scope of the Conformance Program	5
50	1.2	Notation.....	5
51	2	Conformance Clause.....	6
52	2.1	SAML Specification Set	6
53	2.2	Declaration of SAML Conformance	6
54	2.3	Mandatory/Optional Elements in SAML Conformance	8
55	2.4	Impact of Extensions on SAML Conformance	9
56	2.5	Maximum Values of Unbounded Elements.....	9
57	3	Conformance Process.....	11
58	3.1	Implementation and Application Conformance	11
59	3.2	Process for Declaring Conformance.....	12
60	4	Technical Requirements for SAML Conformance.....	13
61	4.1	Test Group 1 – SOAP over HTTP Protocol Binding	13
62	4.1.1	Test Case 1-1: SOAP Binding: Implementation-Under-Test Produces Valid Authentication Assertion in Valid Response to Authentication Query	13
63	4.1.2	Test Case 1-2: SOAP Binding: Implementation-Under-Test Consumes Valid Authentication Assertion, Requested in Valid Authentication Query	14
64	4.1.3	Test Case 1-3: SOAP Binding: Implementation-Under-Test Produces Valid Attribute Assertion in Valid Response to Attribute Query.....	14
65	4.1.4	Test Case 1-4: SOAP Binding: Implementation-Under-Test Consumes Valid Attribute Assertion, Requested in Valid Attribute Query	14
66	4.1.5	Test Case 1-5: SOAP Binding: Implementation-Under-Test Produces Valid Authorization Decision Assertion in Valid Response to Authorization Decision Query	15
67	4.1.6	Test Case 1-6: SOAP Binding: Implementation-Under-Test Consumes Valid Authorization Decision Assertion, Requested in Valid Authorization Decision Query	15
68	4.1.7	Test Case 1-7: SOAP Binding: Implementation-Under-Test Produces Valid Assertions in Valid Response to AssertionIDReference Request.....	15
69	4.1.8	Test Case 1-8: SOAP Binding: Implementation-Under-Test Consumes Valid Assertions, Requested in Valid AssertionIDReference Request.....	16
70	4.2	Test Group 2 – Web Browser SSO Profiles	16
71	4.2.1	Test Case 2-1: Browser/Artifact Profile: Valid Assertions Produced in Response to Valid AssertionArtifact Request.....	16
72	4.2.2	Test Case 2-2: Browser/Artifact Profile: Valid Assertions Request Corresponding to Valid Artifacts Sent in Valid HTTP Message	16
73	4.2.3	Test Case 2-3: Browser/POST Profile: Valid Assertions Received in Valid HTTP POST	17
74	4.2.4	Test Case 2-4: Browser/Post Profile: Valid Assertions Sent in Valid HTTP POST	17

85 5 Test Suite 18
86 6 Conformance Services 19
87 7 References 20
88 Appendix A. Acknowledgments 21
89 Appendix B. Notices 22
90

91 1 Introduction

92 This document describes the program and technical requirements for the SAML conformance system.

93 1.1 Scope of the Conformance Program

94 SAML deals with a rich set of functionalities ranging from assertions about acts of authentication to
95 assertions for policy enforcement. Not all implementers will choose to implement all aspects of the SAML
96 specifications. In order to achieve compatibility and interoperability, applications and software need to be
97 measured for conformance in a uniform manner. The SAML conformance effort aims at fulfilling this need.

98 The deliverables of the SAML conformance effort include:

- 99 • Conformance clause, defining at a high level what conformance means for the SAML standard.
- 100 • Conformance program specification, defining how an implementation or application establishes
101 conformance.
- 102 • Input to the creation of a conformance test suite. This is a high-level specification for a set of test
103 programs, result files, and report generation tools that can be used by vendors of SAML-compliant
104 software, buyers interested in confirming SAML compliance of software, and testing labs running
105 conformance tests on behalf of vendors or buyers.

106 Section 2 of this document provides the SAML Conformance Clause. Section 3 deals with defining and
107 specifying the process by which conformance to the SAML specification set can be demonstrated and
108 certified. Section 4 elucidates the technical requirements that constitute conformance; this includes both
109 the levels of conformance that can be demonstrated and the requirements for each of those levels of
110 conformance. Section 5 describes what a test suite for SAML should include. Section 6 defines the
111 services that may become available to assist in establishing conformance. Section 7 gives information for
112 documents referenced in this specification.

113 1.2 Notation

114 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD
115 NOT", "RECOMMENDED", "DOES", and "OPTIONAL" in this specification are to be interpreted as
116 described in IETF RFC 2119 [**RFC2119**]:

117 ...they **MUST** only be used where it is actually required for interoperation or to limit behavior
118 which has potential for causing harm (e.g., limiting retransmissions)...

119 These keywords are thus capitalized when used to unambiguously specify requirements over protocol and
120 application features and behavior that affect the interoperability and security of implementations. When
121 these words are not capitalized, they are meant in their natural-language sense.

122 2 Conformance Clause

123 The objectives of the SAML Conformance Clause are to:

- 124 • Ensure a common understanding of conformance and what is required to claim conformance
- 125 • Promote interoperability in the exchange of authentication and authorization information
- 126 • Promote uniformity in the development of conformance tests

127 The SAML Conformance Clause explicitly specifies all of the requirements that have to be satisfied to
128 claim conformance to the SAML standard.

129 2.1 SAML Specification Set

130 The following four specifications, in addition to this SAML conformance program specification, comprise
131 the Version 1.1 specification set for SAML:

- 132 • Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) [**SAMLCore**]
- 133 • Security Considerations for the OASIS Security Assertion Markup Language (SAML) [**SAMLSec**]
- 134 • Bindings and Profiles for the OASIS Security Assertion Markup Language (SAML) [**SAMLBind**]
- 135 • Glossary for the OASIS Security Assertion Markup Language (SAML) [**SAMLGloss**]

136 The SAML Core document also references the schema definitions for SAML assertions and protocols:

- 137 • Assertion schema [**SAMLAssertion**]
- 138 • Protocol schema [**SAMLProtocol**]

139 Although additional documents might use or reference the SAML standard (such as white papers,
140 descriptions of custom profiles, and position papers referencing particular issues), they do not constitute
141 part of the standard.

142 2.2 Declaration of SAML Conformance

143 Conformance to the SAML standard can be declared either for the entire standard or for a subset of the
144 standard, based on the requirements that a given implementation or application claims to meet. That is,
145 requirements can be applied at varying levels, so that a given implementation or application of the SAML
146 standard can achieve clearly defined conformance with all or part of the entire set of specifications.

147 SAML conformance **MUST** be expressed in terms of which SAML bindings and profiles are supported by
148 a given application or implementation. The application or implementation claiming conformance to the
149 SAML standard **MUST** support the SOAP protocol binding for assertions containing at least one statement
150 type. An application or implementation **MAY** also support the web browser profiles.

151 For any binding for which an application or implementation claims conformance, the level of conformance
152 **MUST** then be specified in each of these dimensions:

- 153 • Whether the application or implementation acts as producer, consumer, or both producer and
154 consumer of the SAML messages in the supported bindings and profiles.
- 155 • Which assertions and statements the application or implementation supports for each supported
156 binding.

157 Table 1 shows the protocols, protocol bindings, and profiles applicable to each SAML assertion/statement
 158 type. For each SAML binding or profile to which an application or implementation claims conformance, the
 159 claim MUST stipulate whether the producer and/or consumer roles are supported and for which assertions
 160 and statements for those roles.

161 Note that the OASIS Web Services Security Technical Committee has produced a draft “SAML token
 162 profile” of the WSS specification [WSS-SAML], which describes how to use SAML assertions to secure a
 163 web service message. This specification does not discuss conformance to that profile of SAML.

164 For example, an implementation consisting solely of an authentication authority responsible for generating
 165 assertions containing authentication statements and returning those assertions in response to a SOAP-
 166 over-HTTP request for assertion would correspond to the “producer role” for the SOAP over HTTP
 167 binding. If the implementation also supported the return of the assertion in the browser/artifact profile, then
 168 the “producer role” for that profile would also be supported.

169 A SAML protocol <Request> element may contain any one of <AuthenticationQuery>,
 170 <AttributeQuery>, or <AuthorizationDecisionQuery> elements, or, it may contain any number
 171 of <AssertionIDReference> or <AssertionArtifact> elements. For convenience, this document
 172 refers to a SAML request with an <AuthenticationQuery> element as an “authentication query”, a
 173 request with an <AttributeQuery> element as an “attribute query”, and a request with an
 174 <AuthorizationDecisionQuery> element as an “authorization decision query”. SAML requests
 175 containing <AssertionIDReference> or <AssertionArtifact> elements are referred to simply as
 176 requests of those types.

177

178

Table 1: Protocol Bindings and Profiles for SAML Assertions

Binding or Profile	Consumer Role	Producer Role
SOAP over HTTP protocol binding	Send an authentication query to solicit an assertion containing an authentication statement from a producer; consume the returned response and assertion.	Produce an assertion containing an authentication statement and return a response containing the assertion to the consumer.
	Send an attribute query to solicit an assertion containing an attribute statement from a producer; consume the returned assertion.	Produce an assertion containing an attribute statement and return a response containing the assertion to the consumer.
	Send an authorization decision query to solicit an assertion containing an authorization decision statement from a producer; consume the returned assertion.	Produce an assertion containing an authorization decision statement and return a response containing the assertion to the consumer.
	Send an <AssertionIDReference> request to solicit one or more assertions with the associated assertion identifiers from a producer; consume the returned assertions.	Produce a response containing existing assertions with the requested assertion identifiers; send response to the consumer.
Browser/Artifact Profile	Receive one or more artifacts; send an <AssertionArtifact> request; ensure that returned assertions	Produce assertions including an SSO assertion and send corresponding artifacts to a consumer; on receiving

	include a single sign-on assertion; consume the returned assertions.	an <AssertionArtifact> request, produce a response containing the associated assertions; send response to the consumer.
Browser/POST Profile	Receive a response message containing one or more assertions including an SSO assertion in a POST message and consume the assertions.	Produce assertions including an SSO assertion; produce a response message containing the assertions; transfer the response to a consumer via a POST message

179

180 An application or implementation should express its level of conformance in terminology such as the
181 following:

182 [Application or implementation] as both producer and consumer supports all SAML protocol
183 bindings and profiles, for all assertions, statements, and required elements. No optional
184 elements for the assertions, statements, bindings, and profiles are produced.

185 [Application or implementation] as both producer and consumer supports the SOAP protocol
186 binding for all queries, assertions, and statements. It produces the <Conditions> optional
187 elements for all assertions in the SOAP protocol binding. It does not support the browser
188 profiles for any assertion.

189 [Application or implementation] as both producer and consumer supports the SOAP protocol
190 binding for all assertions and statements. It also supports the browser/artifact profile and all
191 required elements. No optional elements for the assertions, statements, bindings, and profiles
192 are produced.

193 An application or implementation that claims conformance for a particular binding or profile MUST support
194 all required elements of that binding or profile and of the assertions supported with that binding or profile.
195 It MUST also state which assertions and statements are supported and which, if any, optional elements for
196 that binding or profile and corresponding assertions and statements are supported.

197 2.3 Mandatory/Optional Elements in SAML Conformance

198 The SOAP protocol binding MUST be implemented by all implementations or applications claiming SAML
199 conformance, for each assertion and statement type claimed as supported through a binding or profile.

200 The SAML schema and binding specifications include both mandatory and optional elements. A
201 conforming application or implementation MUST be able to handle all valid SAML elements, including
202 those that are optional. However, it does not have to produce those optional elements.

203 For example:

- 204 • An application or implementation that consumes assertions must be able to handle assertions that
205 include the optional <Condition> element, such as by rejecting any conditions that it does not
206 recognize.
- 207 • An application or implementation that produces assertions may, but is not required to, include the
208 optional <Condition> element in those assertions.
- 209 • An application or implementation claiming support for an assertion must support the SOAP over HTTP
210 protocol binding. It can also, optionally, implement the protocol by means of another binding.

211 The test cases for SAML conformance are intended to check for support of all valid SAML elements. They
212 also check whether an implementation or application accepts and properly handles optional assertion
213 elements (such as <Condition>) whose value the implementation or application does not recognize.

214 **2.4 Impact of Extensions on SAML Conformance**

215 SAML supports extensions to assertions, statements, protocols, protocol bindings, and profiles. An
216 application or implementation MAY claim conformance to SAML only if its extensions (if any) meet the
217 following requirements:

- 218 • Extensions MUST NOT re-define semantics for existing functions.
 - 219 • Extensions MUST NOT alter the specified behavior of interfaces defined in the SAML specification
220 set.
 - 221 • Extensions MAY add additional behaviors.
 - 222 • Extensions MUST NOT cause standard-conforming functions (i.e., functions that do not use the
223 extensions) to execute incorrectly.
- 224 SAML bindings and profiles MAY be extended so long as the above conditions are met. If a system is
225 extending SAML assertions or statements:
- 226 • The mechanism for determining application conformance and the extensions MUST be clearly
227 described in the documentation, and the extensions MUST be marked as such;
 - 228 • Extensions MUST follow the spirit, principles, and guidelines of the SAML specification set, that is, the
229 specifications MUST be extended in a standard manner as defined in the extension fields.
 - 230 • In the case where an implementation has added additional behaviors, the implementation MUST
231 provide a mechanism whereby a conforming application shall be recognized as such, and be
232 executed in an environment that supports the functional behavior defined in this specification set.

233 Extensions are outside the scope of conformance. There are no mechanisms specified to validate and
234 verify the extensions.

235 **2.5 Maximum Values of Unbounded Elements**

236 The SAML schema supports a number of elements that can be specified multiple times in an assertion,
237 request or response. An application or implementation claiming conformance MUST support at least the
238 values listed in Table 2 below for each of the elements defined as “unbounded” in the SAML schema. In
239 those cases where the maximum value is greater than the listed values, the application or implementation
240 SHOULD state what that maximum supported value is.

241 However, some of the elements in the table can be nested, such that repeated elements have a
242 multiplicative effect on the number of elements. For example, trees of nested unbounded elements
243 include the following:

- 244 Response > Assertion > Statement (of various types)
- 245 Response > Assertion > Advice > Assertion
- 246 Response > Assertion > Conditions > AudienceRestrictionCondition > Audience
- 247 Response > Assertion > Statement > SubjectConfirmation > ConfirmationMethod
- 248 Response > Assertion > AttributeStatement > Attribute > AttributeValue

249 In a response containing 10 assertions, each with 10 AttributeStatements, each with 10 Attributes, each
250 with 10 AttributeValues, this tree alone comprises 10,000 elements.

251 Therefore, in order to minimize the potential impact of nested unbounded elements, an application or
252 implementation MAY limit the total number of elements supported in a given request, response or (when
253 this is used in the POST profile) assertion to no more than 1000 total elements and still claim
254 conformance to the SAML V1.1 specification set.

Table 2: Unbounded Elements

Element	Parent Element	Maximum Value
Statement (various types)	Assertion	1000
DoNotCacheCondition	Conditions	1000
AudienceRestrictionCondition	Conditions	1000
Audience	AudienceRestrictionCondition	1000
AssertionIDReference	Advice	1000
Assertion	Advice	1000
ConfirmationMethod	SubjectConfirmation	1000
AuthorityBinding	AuthenticationStatement	1000
Attribute	AttributeStatement	1000
AttributeValue	Attribute	1000
Action	AuthorizationDecisionStatement	1000
AssertionIDReference	Evidence	1000
Assertion	Evidence	1000
RespondWith	Request	1000
AssertionIDReference	Request	1000
AssertionArtifact	Request	1000
AttributeDesignator	AttributeQuery	1000
Action	AuthorizationDecisionQuery	1000
Assertion	Response	1000

257

3 Conformance Process

258 As discussed in the article “What is this thing called conformance” [NIST/ITL], conformance can comprise
259 any of several levels of formal process:

- 260 • **Conformance testing** (also called conformity assessment) is the execution of automated or non-
261 automated scripts, processes, or other mechanisms to determine whether an application or
262 implementation of a specification deviates from that specification. Conformance testing performed by
263 implementors early on in the development process can find and correct their errors before the
264 software reaches the marketplace, without necessarily being part of either a validation or a
265 certification process.
- 266 • **Validation** is the process of testing software for compliance with applicable specifications or
267 standards. The validation process consists of the steps necessary to perform the conformance testing
268 by using an official test suite in a prescribed manner.
- 269 • **Certification** is the acknowledgment that a validation has been completed and the criteria established
270 by the certifying organization for issuing a certificate have been met. Successful completion of
271 certification results in the issuance of a certificate (or brand) indicating that the implementation
272 conforms to the appropriate specification. It is important to note that certification cannot exist without
273 validation, but validation can exist without certification.

274 The conformance process for SAML is based on validation rather than certification. That is, no certifying
275 organization has been established with the responsible for issuing a statement of conformance with regard
276 to an application or implementation. Therefore, an implementor who has validated SAML conformance by
277 means of conformance testing **MUST NOT** use the term “certified for SAML conformance”. Until and if a
278 certification process is in place, vendor declaration of validation will be the only means of asserting that
279 conformance testing has been performed.

280 The conformance process does not stipulate whether validation is performed by the implementor, by a
281 third party, or by the customer of an application or implementation. Rather, the conformance process
282 describes the way in which conformance testing should be done in order to demonstrate that an
283 application or implementation correctly performs the functionality specified in the standard. Validation
284 achieved through the SAML conformance process provides software developers and users assurance and
285 confidence that the product behaves as expected, performs functions in a known manner, and possesses
286 the prescribed interface or format.

287 The Security Services Technical Committee is responsible for generating the materials that allow vendors,
288 customers, and third parties to evaluate software for SAML conformance. These materials include
289 documentation describing test cases, linked to use cases and requirements, included in this specification.

290 The test cases can be used to create a test suite that can be run against an implementation to
291 demonstrate any of the several levels of conformance defined in the conformance clause of the SAML
292 specification. The Security Services Technical Committee is not responsible for developing the test suite
293 nor for testing of particular implementations.

3.1 Implementation and Application Conformance

294 SAML Conformance is applicable to:

- 295 • Implementations of SAML assertions, statements, protocols and bindings. These could be in the form
296 of toolkits, products incorporating SAML components, or reference implementations that demonstrate
297 the use of SAML components.
298

- 299 • Applications that produce or consume SAML protocol bindings or that execute on SAML
300 implementations (for example, using a SAML toolkit to support multi-domain single sign-on)

301 A conforming **implementation** MUST meet all the following criteria:

- 302 1. The implementation MUST support all the required interfaces defined within the specification set for a
303 given binding or profile. It MUST also specify which assertions and statements relevant to that binding
304 or profile are supported. The implementation MUST support the functional behavior described in the
305 specification.
- 306 2. The implementation MAY provide additional or enhanced facilities not required by this specification
307 set. These nonstandard extensions MUST NOT alter the specified behavior of interfaces defined in
308 this specification. They MAY add additional behaviors. In these circumstances, the implementation
309 MUST provide a mechanism whereby a SAML conforming application shall be recognized as such,
310 and be executed in an environment that supports the functional behavior defined in this specification
311 set.

312 A conforming **application** MUST meet all the following criteria:

- 313 1. The application MUST be able to execute on any conforming implementation.
- 314 2. If an application requires a particular feature set that is not available on a specific implementation,
315 then the application MUST act within the bounds of the SAML specification set, even though that
316 means that the application does not perform any useful function. Specifically, the application MUST
317 do no harm, and MUST correctly return resources and vacate memory upon discovery that a required
318 element is not present.

319 **3.2 Process for Declaring Conformance**

320 The following process is to be followed in declaring that an application or implementation conforms to the
321 SAML standard:

- 322 1. Determine which bindings and protocols will be asserted as conforming.
- 323 2. Implement the test suite for the conformance tests relevant to the conformance being claimed.
- 324 3. Validate the application or implementation by executing those conformance tests.
- 325 4. Send the statement claiming conformance to the Security Services Technical Committee so that it can
326 be posted on the SAML web site. A statement of any bindings and profiles being used that are not part
327 of the SAML standard should also be sent to the Security Services Technical Committee at the same
328 time for posting on the SAML web site.

329

4 Technical Requirements for SAML Conformance

330 This section defines the technical criteria that apply to declaring conformance to the SAML standard. The
331 requirements are specified as test cases, corresponding to the 12 possible subsets of conformance
332 defined in Table 1.

333 Each test case includes:

- 334 • A description of the test purpose (that is, what is being tested – the conditions, requirements, or
335 capabilities which are to be addressed by a particular test)
- 336 • The pass/fail criteria
- 337 • A reference to the requirement in the requirements document relevant to the test case
- 338 • A reference to the section in the specification set from which the test case is derived (that is,
339 traceability back to the specification)

340 For each assertion and statement type, both required tests for producing and consuming the assertion, as
341 well as tests related to protocols, bindings, and profiles, are specified.

342 4.1 Test Group 1 – SOAP over HTTP Protocol Binding

343 The test cases in this test group check for conformance to the SAML SOAP protocol binding. Any
344 implementation or application claiming conformance to SAML MUST be able to execute these test cases
345 successfully for the claimed assertion or assertions and role (producer or consumer), even if support for
346 this protocol binding is incidental to the primary purposes of the application or implementation.

347 For convenience, assertions containing an authentication statement will be referred to in this section as
348 *authentication assertions*, assertions containing an attribute statement as *attribute assertions*, and
349 assertions containing an authorization decision statement as *authorization decision assertions*.

350

351 4.1.1 Test Case 1-1: SOAP Binding: Implementation-Under-Test Produces 352 Valid Authentication Assertion in Valid Response to Authentication 353 Query

354 **Description:** This test case requests and receives an authentication assertion created by an
355 implementation-under-test using an authentication query in the SOAP binding. It then confirms that the
356 authentication assertion returned by the implementation-under-test is valid for all required functionality.

357 **Pass/Fail Criteria:** The authentication assertion contains all required elements in the correct format and
358 sequence, the authentication query is accepted by implementation-under-test, and the response contains
359 all required elements in correct sequence.

360 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

361 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

362 **Implementation Notes:** The implementation-under-test executes the authentication assertion producer
363 role.

364 **4.1.2 Test Case 1-2: SOAP Binding: Implementation-Under-Test Consumes**
365 **Valid Authentication Assertion, Requested in Valid Authentication Query**

366 **Description:** This test case receives an authentication query created by an implementation-under-test in
367 the SOAP binding. It confirms that the authentication query is valid for all required functionality. The test
368 case returns an authentication assertion and confirms that the assertion is consumed.

369 **Pass/Fail Criteria:** The authentication query contains all required elements in the correct format and
370 sequence; the authentication response and assertion are consumed.

371 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

372 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

373 **Implementation Notes:** The implementation-under-test executes the authentication assertion consumer
374 role. It is up to the test program and implementation-under-test to determine how to validate that the
375 assertion was consumed.

376 **4.1.3 Test Case 1-3: SOAP Binding: Implementation-Under-Test Produces**
377 **Valid Attribute Assertion in Valid Response to Attribute Query**

378 **Description:** This test case requests and receives an attribute assertion created by an implementation-
379 under-test using an attribute query in the SOAP binding. It then confirms that the attribute assertion
380 returned by the implementation-under-test is valid for all required functionality.

381 **Pass/Fail Criteria:** The attribute assertion contains all required elements in the correct format and
382 sequence, the attribute query is accepted by implementation-under-test, and the response contains all
383 required elements in correct sequence.

384 **Requirements Reference:** R-AUTHZ and R-MULTIDOMAIN

385 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

386 **Implementation Notes:** The implementation-under-test executes the attribute assertion producer role.

387 **4.1.4 Test Case 1-4: SOAP Binding: Implementation-Under-Test Consumes**
388 **Valid Attribute Assertion, Requested in Valid Attribute Query**

389 **Description:** This test case receives an attribute query sent by an implementation-under-test in the SOAP
390 binding. It confirms that the attribute query is valid for all required functionality. The test case then returns
391 an attribute assertion and confirms that the assertion is consumed.

392 **Pass/Fail Criteria:** The attribute query contains all required elements in the correct format and sequence;
393 attribute response and assertion are consumed.

394 **Requirements Reference:** R-AUTHZ and R-MULTIDOMAIN

395 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

396 **Implementation Notes:** The implementation-under-test executes the attribute assertion consumer role. It
397 is up to the test program and implementation-under-test to determine how to validate that assertion was
398 consumed.

399 **4.1.5 Test Case 1-5: SOAP Binding: Implementation-Under-Test Produces**
400 **Valid Authorization Decision Assertion in Valid Response to**
401 **Authorization Decision Query**

402 **Description:** This test case requests and receives an authorization decision assertion created by an
403 implementation-under-test using an authorization decision query in the SOAP binding. It then confirms
404 that the authorization decision assertion returned by the implementation-under-test is valid for all required
405 functionality.

406 **Pass/Fail Criteria:** The authorization decision assertion contains all required elements in the correct
407 format and sequence, the authorization decision query is accepted by implementation-under-test, and the
408 response contains all required elements in correct sequence.

409 **Requirements Reference:** R-AUTHZDECISION and R-MULTIDOMAIN

410 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

411 **Implementation Notes:** The implementation-under-test executes the authorization decision assertion
412 producer role.

413 **4.1.6 Test Case 1-6: SOAP Binding: Implementation-Under-Test Consumes**
414 **Valid Authorization Decision Assertion, Requested in Valid Authorization**
415 **Decision Query**

416 **Description:** This test case receives an authorization decision query created by an implementation-under-
417 test in the SOAP binding. It confirms that the received authorization decision query is valid for all required
418 functionality. It returns an authorization decision assertion to the implementation-under-test and confirms
419 that the assertion is consumed.

420 **Pass/Fail Criteria:** The authorization decision query contains all required elements in the correct format
421 and sequence; authorization decision response and assertion are consumed.

422 **Requirements Reference:** R-AUTHZDECISION and R-MULTIDOMAIN

423 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

424 **Implementation Notes:** The implementation-under-test executes the authorization decision assertion
425 consumer role. It is up to the test program and implementation-under-test to determine how to validate
426 that assertion was consumed.

427 **4.1.7 Test Case 1-7: SOAP Binding: Implementation-Under-Test Produces**
428 **Valid Assertions in Valid Response to AssertionIDReference Request**

429 **Description:** This test case requests and receives assertions created by an implementation-under-test
430 using an AssertionIDReference request in the SOAP binding. It then confirms that the assertions returned
431 by the implementation-under-test are valid for all required functionality.

432 **Pass/Fail Criteria:** The returned assertions contain all required elements in the correct format and
433 sequence, the AssertionIDReference request is accepted by implementation-under-test, and the response
434 contains all required elements in correct sequence.

435 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

436 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

437 **Implementation Notes:** The implementation-under-test executes the assertion producer role.

438 **4.1.8 Test Case 1-8: SOAP Binding: Implementation-Under-Test Consumes**
439 **Valid Assertions, Requested in Valid AssertionIDReference Request**

440 **Description:** This test case receives an AssertionIDReference request in the SOAP binding created by an
441 implementation-under-test. It confirms that the received AssertionIDReference request is valid for all
442 required functionality. The test case returns the requested assertions and confirms that the assertions are
443 consumed.

444 **Pass/Fail Criteria:** The AssertionIDReference request contains all required elements in the correct format
445 and sequence; the response and assertions are consumed.

446 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

447 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 3.1

448 **Implementation Notes:** The implementation-under-test executes the assertion consumer role. It is up to
449 the test program and implementation-under-test to determine how to validate that assertions were
450 consumed.

451 **4.2 Test Group 2 – Web Browser SSO Profiles**

452 The test cases in this test group check for conformance to the web browser single sign-on (SSO) profiles
453 of the SAML standard. Both the browser/artifact and browser/POST profiles are optional. Any
454 implementation or application claiming conformance to the browser/artifact profile MUST be able to
455 execute Test Case 2-1 successfully for the assertion producer role and/or Test Case 2-2 successfully for
456 the assertion consumer role. Any implementation or application claiming conformance to the
457 browser/POST profile MUST be able to execute Test Case 2-3 successfully for the assertion producer role
458 and/or Test Case 2-4 successfully for the assertion consumer role.

459 **4.2.1 Test Case 2-1: Browser/Artifact Profile: Valid Assertions Produced in**
460 **Response to Valid AssertionArtifact Request**

461 **Description:** This test case receives artifacts in a valid HTTP message from an implementation-under-
462 test. The test case confirms that the artifacts are valid for all required functionality. It then uses the
463 AssertionArtifact request in the SOAP binding to request and receive assertions created by an
464 implementation-under-test corresponding to the artifacts. It then confirms that the returned assertions
465 include an SSO assertion and is valid for all required functionality.

466 **Pass/Fail Criteria:** .Received artifacts have expected formats. AssertionArtifact request contains all
467 required elements in correct format and sequence and is accepted by the implementation-under-test; An
468 assertion is returned for every artifact in the AssertionArtifact request. Returned assertions include an
469 SSO assertion.

470 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

471 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 4.1.1

472 **Implementation Notes:** Test program performs the destination site (consumer) operations for the profile;
473 implementation-under-test performs source site (producer) operations.

474 **4.2.2 Test Case 2-2: Browser/Artifact Profile: Valid Assertions Request**
475 **Corresponding to Valid Artifacts Sent in Valid HTTP Message**

476 **Description:** This test case sends valid artifacts in a valid HTTP message to an implementation-under-
477 test. The test case then receives an AssertionArtifact request containing the artifacts from the

478 implementation-under-test. It confirms that the AssertionArtifact request is valid for all required
479 functionality, then returns the requested assertions to the implementation-under-test, and confirms that the
480 assertion was consumed.

481 **Pass/Fail Criteria:** AssertionArtifact request contains all required elements in the correct format and
482 sequence.

483 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

484 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 4.1.1

485 **Implementation Notes:** Test program performs the source site (producer) operations for the profile;
486 implementation-under-test performs destination site (consumer) operations.

487 **4.2.3 Test Case 2-3: Browser/POST Profile: Valid Assertions Received in** 488 **Valid HTTP POST**

489 **Description:** This test case receives an HTTP POST message from an implementation-under-test
490 containing a SAML protocol response message with one or more assertions and including an SSO
491 assertion and checks that the assertions are valid.

492 **Pass/Fail Criteria:** SSO assertion sent by implementation-under-test MUST contain all required
493 information in the right sequence and format. Any optional information included (including conditions)
494 MUST NOT compromise the validity of the required information.

495 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

496 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 4.1.2

497 **Implementation Notes:** Test program (consumer role) implementing this test case establishes
498 successful execution of the test case by inspection of the format of the returned assertion.

499 **4.2.4 Test Case 2-4: Browser/Post Profile: Valid Assertions Sent in Valid** 500 **HTTP POST**

501 **Description:** This test case sends a SAML protocol response message in an HTTP POST message to an
502 implementation-under-test containing an SSO and other assertions and checks that the assertions are
503 consumed.

504 **Pass/Fail Criteria:** Implementation-under-test allows access based on assertions it receives and
505 consumes.

506 **Requirements Reference:** R-AUTHN and R-MULTIDOMAIN

507 **Specification Reference:** [SAMLCore] Sections 2.3, 2.4, and 3; [SAMLBind] Section 4.1.2

508 **Implementation Notes:** It is up to the test program and implementation-under-test to determine how to
509 validate that assertion was consumed.

510

5 Test Suite

511 A test suite, which is the combination of test cases and test documentation, is used to check whether an
512 implementation or application satisfies the requirements in the standard. The test cases, implemented by
513 a test tool or a set of files (such as data, programs, scripts, or instructions for manual action), check each
514 requirement in the specification to determine whether the results produced by the implementation or
515 application match the expected results, as defined by the specification.

516 The test documentation describes how the testing is to be done and the directions for the tester to follow.
517 Additionally, the documentation should be detailed enough so that testing of a given implementation can
518 be repeated with no change in test results.

519 Conformance testing is black-box testing to test the functionality of an implementation. This means that
520 the internal structure or the source code of a candidate implementation is not available to the tester.
521 However, content and format of received or returned messages can be inspected as part of the
522 determination of conformance.

523 Any test suite for SAML should consist of platform independent, non-biased, objective tests. Generally, a
524 conformance test suite is a collection of combinations of legal and illegal inputs to the implementation
525 being tested, together with a corresponding collection of expected results. Only the requirements
526 specified in the standard are testable. A test suite should not check any implementation properties that
527 are not described by the standard or set of standards. A test suite cannot require features that are optional
528 in a standard, but if such features are present, a test suite could include tests for those features. A test
529 suite does not assess the performance of an implementation unless performance requirements are
530 specified in the specification, although implementation dependencies or machine dependencies can be
531 demonstrated through the execution of the test cases.

532 The results of conformance testing apply only to the implementation and environment for which the tests
533 are run. Test suites can be provided as a web-based system executed on a remote server, downloadable
534 files for local execution, or a combination of remote and local access and execution. The method for
535 providing and delivering the test suite depends on what is being tested as well as the objective for test
536 suite use – that is, providing self-test capability or formal certification testing.

537

6 Conformance Services

538 The OASIS Security Services Technical Committee does not itself provide conformance services. As
539 SAML test suites become available and experience with SAML identified appropriate conformance testing
540 approaches, the Conformance Specification will describe the services which a conformance services
541 organization should provide, including software services, releases, self-test kit, actual computer systems,
542 facilities, web based interfaces, and availability.

543

7 References

- 544 **[NIST/ITL]** “*What is this thing called conformance*” [Rosenthal, Brady; NIST/ITL Bulletin,
545 January 2001] [http://www.itl.nist.gov/div897/ctg/conformance/bulletin-](http://www.itl.nist.gov/div897/ctg/conformance/bulletin-conformance.htm)
546 [conformance.htm](http://www.itl.nist.gov/div897/ctg/conformance/bulletin-conformance.htm).
- 547 **[RFC2119]** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
548 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 549 **[SAMLAssertion]** E. Maler et al., *Assertions Schema for the OASIS Security Assertion Markup*
550 *Language (SAML)*, <http://www.oasis-open.org/committees/security/>, OASIS, May
551 2003.
- 552 **[SAMLBind]** E. Maler et al., *Bindings and Profiles for the OASIS Security Assertion Markup*
553 *Language (SAML)*, <http://www.oasis-open.org/committees/security/>, OASIS, May
554 2003.
- 555 **[SAMLCore]** E. Maler et al., *Assertions and Protocol for the OASIS Security Assertion Markup*
556 *Language (SAML)*, <http://www.oasis-open.org/committees/security/>, OASIS, May
557 2003.
- 558 **[SAMLGloss]** E. Maler et al., *Glossary for the OASIS Security Assertion Markup Language*
559 *(SAML)*, <http://www.oasis-open.org/committees/security/>, OASIS, May 2003.
- 560 **[SAMLProtocol]** E. Maler et al., *Protocol Schema for the OASIS Security Assertion Markup*
561 *Language (SAML)*, <http://www.oasis-open.org/committees/security/>, OASIS, May
562 2003.
- 563 **[SAMLSec]** E. Maler et al., *Security Considerations for the OASIS Security Assertion Markup*
564 *Language (SAML)*, <http://www.oasis-open.org/committees/security/>, OASIS, May
565 2003.
- 566 **[WSS-SAML]** P. Hallam-Baker et al., *Web Services Security: SAML Token Profile*, OASIS,
567 March 2003, <http://www.oasis-open.org/committees/wss>.

568 **Appendix A. Acknowledgments**

569 The editors would like to acknowledge the contributions of the OASIS SAML Technical Committee, whose
570 voting members at the time of publication were:

- 571 • Frank Siebenlist, Argonne National Laboratory
- 572 • Irving Reid, Baltimore Technologies
- 573 • Hal Lockhart, BEA Systems
- 574 • Steven Lewis, Booz Allen Hamilton
- 575 • John Hughes, Entegriy Solutions
- 576 • Carlisle Adams, Entrust
- 577 • Jason Rouault, HP
- 578 • Maryann Hondo, IBM
- 579 • Anthony Nadalin, IBM
- 580 • Scott Cantor, Individual
- 581 • Bob Morgan, Individual
- 582 • Trevor Perrin, Individual
- 583 • Padraig Moloney, NASA
- 584 • Prateek Mishra, Netegrity (co-chair)
- 585 • Frederick Hirsch, Nokia
- 586 • Senthil Sengodan, Nokia
- 587 • Timo Skytta, Nokia
- 588 • Charles Knouse, Oblix
- 589 • Steve Anderson, OpenNetwork
- 590 • Simon Godik, OverXeer
- 591 • Rob Philpott, RSA Security (co-chair)
- 592 • Dipak Chopra, SAP
- 593 • Jahan Moreh, Sigaba
- 594 • Bhavna Bhatnagar, Sun Microsystems
- 595 • Jeff Hodges, Sun Microsystems
- 596 • Eve Maler, Sun Microsystems (coordinating editor)
- 597 • Emily Xu, Sun Microsystems
- 598 • Phillip Hallam-Baker, VeriSign

Appendix B. Notices

600 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
601 might be claimed to pertain to the implementation or use of the technology described in this document or
602 the extent to which any license under such rights might or might not be available; neither does it represent
603 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to
604 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made
605 available for publication and any assurances of licenses to be made available, or the result of an attempt
606 made to obtain a general license or permission for the use of such proprietary rights by implementors or
607 users of this specification, can be obtained from the OASIS Executive Director.

608 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or
609 other proprietary rights which may cover technology that may be required to implement this specification.
610 Please address the information to the OASIS Executive Director.

611 **Copyright © OASIS Open 2003. All Rights Reserved.**

612 This document and translations of it may be copied and furnished to others, and derivative works that
613 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and
614 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and
615 this paragraph are included on all such copies and derivative works. However, this document itself does
616 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as
617 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights
618 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it
619 into languages other than English.

620 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors
621 or assigns.

622 This document and the information contained herein is provided on an "AS IS" basis and OASIS
623 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY
624 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR
625 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.