

# Web Services Security: SAML Token Profile 1.1

## Draft 11, 7 November 2005

### Document Identifier:

[wss-v1.1-spec-draft-SAMLTokenProfile-11](#)

### OASIS Identifier:

{WSS: SOAP Message Security }-{SAMLTokenProfile}-{1.1} (OpenOffice) (PDF) (HTML)

### Location:

Persistent: [\[persistent location\]](#)

This Version: <http://docs.oasis-open.org/wss/oasis-wss-SAMLTokenProfile-1.1>

Previous Version:<http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0>

### Technical Committee:

OASIS Web Services Security (WSS) TC

### Chairs:

Kelvin Lawrence	IBM
Chris Kaler	Microsoft

### Editors

Ronald Monzillo	Sun
Chris Kaler	Microsoft
Anthony Nadalin	IBM
Phillip Hallem-Baker	VeriSign

### Abstract:

This document describes how to use Security Assertion Markup Language (SAML) V1.1 and V2.0 assertions with the [Web Services Security \(WSS\): SOAP Message Security V1.1](#) specification.

With respect to the description of the use of SAML V1.1, this document subsumes and is totally consistent with the Web Services Security: SAML Token Profile 1.0 and includes all corrections identified in the 1.0 errata.

### Status:

This document was last revised or approved by the membership of the Web Services Security TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

Technical Committee members should send comments on this specification to the Technical Committee's email list. Others should send comments to the Technical Committee by using the "Send A Comment" button on the Technical Committee's web page at [www.oasis-open.org/committees/wss](http://www.oasis-open.org/committees/wss).

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page ([www.oasis-open.org/committees/wss/ipr.php](http://www.oasis-open.org/committees/wss/ipr.php)).

The non-normative errata for this specification is located at [www.oasis-open.org/committees/wss](http://www.oasis-open.org/committees/wss).

## Notices

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

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

48 Copyright (C) OASIS Open 2002-2005. All Rights Reserved.

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

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

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

63 OASIS has been notified of intellectual property rights claimed in regard to some or all of the contents of  
64 this specification. For more information consult the online list of claimed rights.

## Table of Contents

66	1 Introduction.....	5
67	1.1 Goals.....	5
68	1.1.1 Non-Goals.....	5
69	2 Notations and Terminology.....	6
70	2.1 Notational Conventions.....	6
71	2.2 Namespaces.....	6
72	2.3 Terminology.....	6
73	3 Usage.....	8
74	3.1 Processing Model.....	8
75	3.2 SAML Version Differences.....	8
76	3.2.1 Assertion Identifier.....	8
77	3.2.2 Relationship of Subjects to Statements.....	8
78	3.2.3 Assertion URI Reference Replaces AuthorityBinding.....	10
79	3.2.4 Attesting Entity Identifier.....	10
80	3.3 Attaching Security Tokens.....	10
81	3.4 Identifying and Referencing Security Tokens.....	11
82	3.4.1 SAML Assertion Referenced from Header or Element.....	13
83	3.4.2 SAML Assertion Referenced from KeyInfo.....	14
84	3.4.3 SAML Assertion Referenced from SignedInfo.....	16
85	3.4.4 SAML Assertion Referenced from Encrypted Data Reference.....	17
86	3.4.5 SAML Version Support and Backward Compatability.....	17
87	3.5 Subject Confirmation of SAML Assertions.....	17
88	3.5.1 Holder-of-key Subject Confirmation Method.....	18
89	3.5.2 Sender-vouches Subject Confirmation Method.....	21
90	3.5.3 Bearer Confirmation Method.....	25
91	3.6 Error Codes.....	25
92	4 Threat Model and Countermeasures (non-normative).....	27
93	4.1 Eavesdropping.....	27
94	4.2 Replay.....	27
95	4.3 Message Insertion.....	27
96	4.4 Message Deletion.....	27
97	4.5 Message Modification.....	27
98	4.6 Man-in-the-Middle.....	28
99	5 References .....	29
100	Appendix A. Acknowledgements.....	30
101		

---

# 102 1 Introduction

103 The [WSS: SOAP Message Security](#) specification defines a standard set of [SOAP](#) extensions that  
104 implement SOAP message authentication and encryption. This specification defines the use of Security  
105 Assertion Markup Language (SAML) assertions as security tokens from the `<wsse:Security>` header  
106 block defined by the [WSS: SOAP Message Security](#) specification.

## 107 1.1 Goals

108 The goal of this specification is to define the use of SAML V1.1 and V2.0 assertions in the context of  
109 [WSS: SOAP Message Security](#) including for the purpose of securing [SOAP](#) messages and [SOAP](#)  
110 message exchanges. To achieve this goal, this profile describes how:

- 111 1. SAML assertions are carried in and referenced from `<wsse:Security>` Headers.
- 112 2. SAML assertions are used with XML signature to bind the subjects and statements of the assertions  
113 (i.e., the claims) to a SOAP message.

### 114 1.1.1 Non-Goals

115 The following topics are outside the scope of this document:

- 116 1. Defining SAML statement syntax or semantics.
- 117 2. Describing the use of SAML assertions other than for SOAP Message Security.
- 118 3. Describing the use of SAML V1.0 assertions with the [Web Services Security \(WSS\): SOAP Message](#)  
119 [Security](#) specification.

## 2 Notations and Terminology

120

This section specifies the notations, namespaces, and terminology used in this specification.

121

### 2.1 Notational Conventions

122

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119](#).

123

124

125

This document uses the notational conventions defined in the WS-Security SOAP Message Security document.

126

127

Namespace URIs (of the general form "some-URI") represent some application-dependent or context-dependent URI as defined in [RFC2396](#).

128

129

This specification is designed to work with the general SOAP message structure and message processing model, and should be applicable to any version of SOAP. The current SOAP 1.2 namespace URI is used herein to provide detailed examples, but there is no intention to limit the applicability of this specification to a single version of SOAP.

130

131

132

133

Readers are presumed to be familiar with the terms in the [Internet Security Glossary](#).

134

### 2.2 Namespaces

135

The appearance of the following [XML-ns] namespace prefixes in the examples within this specification should be understood to refer to the corresponding namespaces (from the following table) whether or not an XML namespace declaration appears in the example:

136

137

138

Prefix	Namespace
s11	<a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a>
s12	<a href="http://www.w3.org/2003/05/soap-envelope">http://www.w3.org/2003/05/soap-envelope</a>
ds	<a href="http://www.w3.org/2000/09/xmldsig#">http://www.w3.org/2000/09/xmldsig#</a>
xenc	<a href="http://www.w3.org/2001/04/xmlenc">http://www.w3.org/2001/04/xmlenc</a>
wsse	<a href="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd</a>
wsse11	<a href="http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd">http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd</a>
wsu	<a href="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd</a>
saml	urn: oasis:names:tc:SAML:1.0:assertion
saml2	urn: oasis:names:tc:SAML:2.0:assertion
samlp	urn: oasis:names:tc:SAML:1.0:protocol

Table-1 Namespace Prefixes

139

### 2.3 Terminology

140

This specification employs the terminology defined in the [WSS: SOAP Message Security](#) specification. The definitions for additional terminology used in this specification appear below.

141

142

- 143 Attesting Entity – the entity that provides the confirmation evidence that will be used to establish the  
144 correspondence between the subjects and claims of SAML statements (in SAML assertions) and SOAP  
145 message content.
- 146 Confirmation Method Identifier – the value within a SAML SubjectConfirmation element that identifies the  
147 subject confirmation process to be used with the corresponding statements.
- 148 Subject Confirmation – the process of establishing the correspondence between the subject and claims of  
149 SAML statements (in SAML assertions) and SOAP message content by verifying the confirmation  
150 evidence provided by an attesting entity.
- 151 SAML Assertion Authority - A *system entity* that issues *assertions*.
- 152 Subject – A representation of the entity to which the claims in one or more SAML statements apply.

---

## 153 3 Usage

154 This section defines the specific mechanisms and procedures for using SAML assertions as security  
155 tokens.

### 156 3.1 Processing Model

157 This specification extends the token-independent processing model defined by the [WSS: SOAP Message](#)  
158 [Security](#) specification.

159 When a receiver processes a `<wsse:Security>` header containing or referencing SAML assertions, it  
160 selects, based on its policy, the signatures and assertions that it will process. It is assumed that a  
161 receiver's signature selection policy MAY rely on semantic labeling<sup>1</sup> of  
162 `<wsse:SecurityTokenReference>` elements occurring in the `<ds:KeyInfo>` elements within the  
163 signatures. It is also assumed that the assertions selected for validation and processing will include those  
164 referenced from the `<ds:KeyInfo>` and `<ds:SignedInfo>` elements of the selected signatures.

165 As part of its validation and processing of the selected assertions, the receiver MUST<sup>2</sup> establish the  
166 relationship between the subject and claims of the SAML statements (of the referenced SAML assertions)  
167 and the entity providing the evidence to satisfy the confirmation method defined for the statements (i.e.,  
168 the attesting entity). Two methods for establishing this correspondence, `holder-of-key` and `sender-`  
169 `vouches` are described below. Systems implementing this specification MUST implement the processing  
170 necessary to support both of these subject confirmation methods.

### 171 3.2 SAML Version Differences

172 The following sub-sections describe the differences between SAML V1.1 and V2.0 that apply to this  
173 specification.

#### 174 3.2.1 Assertion Identifier

175 In SAML V1.1 the name of the assertion identifier attribute is "AssertionID". In SAML v2.0 the name of the  
176 assertion identifier attribute is "ID". In both versions the type of the identifier attribute is `xs:ID`.

#### 177 3.2.2 Relationship of Subjects to Statements

178 A SAML assertion contains a collection of 0 or more statements. In SAML V1.1, a separate subject with  
179 separate subject confirmation methods may be specified for each statement of an assertion. In SAML  
180 V2.0, at most one subject and at most one set of subject confirmation methods may be specified for all  
181 the statements of the assertion. These distinctions are described in more detail by the following  
182 paragraphs.

183 A SAML V1.1 statement that contains a `<saml:Subject>` element (i.e., a subject statement) may  
184 contain a `<saml:SubjectConfirmation>` element that defines the rules for confirming the subject and  
185 claims of the statement. If present, the `<saml:SubjectConfirmation>` element occurs within the  
186 subject element, and defines one or more methods (i.e., `<saml:ConfirmationMethod>` elements) by  
187 which the statement may be confirmed and will include a `<ds:KeyInfo>`<sup>3</sup> element when any of the  
188 specified methods are based on demonstration of a confirmation key. The

---

<sup>1</sup> The optional `Usage` attribute of the `<wsse:SecurityTokenReference>` element MAY be used to associate one of more semantic usage labels (as URIs) with a reference and thus use of a Security Token. Please refer to [WSS: SOAP Message Security](#) for the details of this attribute.

<sup>2</sup> When the confirmation method is `urn:oasis:names:tc:SAML:1.0:cm:bearer`, proof of the relationship between the attesting entity and the subject of the statements in the assertion is implicit and no steps need be taken by the receiver to establish this relationship.

<sup>3</sup> When a `<ds:KeyInfo>` element is specified, it identifies the key that applies to all the key confirmed methods of the confirmation element.

189 <saml:SubjectConfirmation> element also provides for the inclusion of additional information to be  
 190 applied in the confirmation method processing via the optional <saml:SubjectConfirmationData>  
 191 element. The following example depicts a SAML V1.1 assertion containing two subject statements with  
 192 different subjects and different subject confirmation elements.

```

193 <saml:Assertion
194   ...
195   <saml:SubjectStatement>
196     <saml:Subject>
197       <saml:NameIdentifier
198         ...
199       </saml:NameIdentifier>
200       <saml:SubjectConfirmation>
201         <saml:ConfirmationMethod>
202           urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
203         </saml:ConfirmationMethod>
204         <saml:ConfirmationMethod>
205           urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
206         </saml:ConfirmationMethod>
207         <ds:KeyInfo>
208           <ds:KeyValue>...</ds:KeyValue>
209         </ds:KeyInfo>
210       </saml:SubjectConfirmation>
211     </saml:Subject>
212   ...
213 </saml:SubjectStatement>
214 <saml:SubjectStatement>
215   <saml:Subject>
216     <saml:NameIdentifier
217       ...
218     </saml:NameIdentifier>
219     <saml:SubjectConfirmation>
220       <saml:ConfirmationMethod>
221         urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
222       </saml:ConfirmationMethod>
223     </saml:SubjectConfirmation>
224   </saml:Subject>
225   ...
226 </saml:SubjectStatement>
227   ...
228 </saml:Assertion>
  
```

229 A SAML V2.0 assertion may contain a single <saml2:Subject> that applies to all the statements of the  
 230 assertion. When a subject is included in A SAML V2.0 assertion, it may contain any number of  
 231 <saml2:SubjectConfirmation> elements, satisfying any of which is sufficient to confirm the subject  
 232 and all the statements of the assertion. Each <saml2:SubjectConfirmation> element identifies a  
 233 single confirmation method (by attribute value) and may include an optional  
 234 <saml2:SubjectConfirmationData> element that is used to specify optional confirmation method  
 235 independent condition attributes and to define additional method specific confirmation data. In the case of  
 236 a key dependent confirmation method, a complex schema type,  
 237 saml2:KeyInfoConfirmationDataType, that includes 1 or more <ds:KeyInfo> elements, can be  
 238 specified as the xsi:type of the <saml2:SubjectConfirmationData> element. In this case, each  
 239 <ds:KeyInfo> element identifies a key that may be demonstrated to confirm the assertion. The following  
 240 example depicts a SAML V2.0 assertion containing a subject with multiple confirmation elements that  
 241 apply to all the statements of the assertion.

```

242 <saml2:Assertion
243   ...
244   <saml2:Subject>
245     <saml2:NameID>
246       ...
247     </saml2:NameID>
248     <saml2:SubjectConfirmation
249       Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
  
```

```

250     <saml2:SubjectConfirmationData>
251       Address="129.148.9.42"
252     </saml2:SubjectConfirmationData>
253   </saml2:SubjectConfirmation>
254   <saml2:SubjectConfirmation
255     Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
256     <saml2:SubjectConfirmationData
257       xsi:type="saml2:KeyInfoConfirmationDataType">
258       <ds:KeyInfo>
259         <ds:KeyValue>...</ds:KeyValue>
260       </ds:KeyInfo>
261     </saml2:SubjectConfirmationData>
262   </saml2:SubjectConfirmation>
263 </saml2:Subject>
264   ...
265 <saml2:Statement>
266   ...
267 </saml2:Statement>
268
269 <saml2:Statement>
270   ...
271 </saml2:Statement>
272   ...
273
274 </saml2:Assertion>

```

### 275 3.2.3 Assertion URI Reference Replaces AuthorityBinding

276 SAML V1.1 defines the (deprecated) `<saml:AuthorityBinding>` element so that a relying party can  
 277 locate and communicate with an assertion authority to acquire a referenced assertion.

278 The `<saml:AuthorityBinding>` element was removed from SAML V2.0. [SAMLBindV2] requires that  
 279 an assertion authority support a URL endpoint at which an assertion will be returned in response to an  
 280 HTTP request with a single query string parameter named ID.

281 For example, if the documented endpoint at an assertion authority is:

```
282 https://saml.example.edu/assertion-authority
```

283 then the following request will cause the assertion with ID "abcde" to be returned:

```
284 https://saml.example.edu/assertion-authority?ID=abcde
```

### 285 3.2.4 Attesting Entity Identifier

286 The `<saml2:SubjectConfirmation>` element of SAML V2.0 provides for the optional inclusion of an  
 287 element (i.e., NameID) to identify the expected attesting entity as distinct from the subject of the assertion.

```

288 <saml2:SubjectConfirmation
289   Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
290   <NameID>
291     gateway
292   </NameID>
293   <saml2:SubjectConfirmationData>
294     Address="129.148.9.42"
295   </saml2:SubjectConfirmationData>
296 </saml2:SubjectConfirmation>

```

## 297 3.3 Attaching Security Tokens

298 SAML assertions are attached to SOAP messages using [WSS: SOAP Message Security](#) by placing  
 299 assertion elements or references to assertions inside a `<wsse:Security>` header. The following  
 300 example illustrates a SOAP message containing a bearer confirmed SAML V1.1 assertion in a  
 301 `<wsse:Security>` header.

```

302 <S12:Envelope>
303   <S12:Header>
304     <wsse:Security>
305
306       <saml:Assertion
307         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
308         IssueInstant="2003-04-17T00:46:02Z"
309         Issuer="www.opensaml.org"
310         MajorVersion="1"
311         MinorVersion="1"
312         . . .
313       <saml:AuthenticationStatement>
314         <saml:Subject>
315           <saml:NameIdentifier
316             NameQualifier="www.example.com"
317             Format="urn:oasis:names:tc:SAML:1.1:nameid-
318 format:X509SubjectName">
319             uid=joe,ou=people,ou=saml-demo,o=baltimore.com
320           </saml:NameIdentifier>
321           <saml:SubjectConfirmation>
322             <saml:ConfirmationMethod>
323               urn:oasis:names:tc:SAML:1.0:cm:bearer
324             </saml:ConfirmationMethod>
325           </saml:SubjectConfirmation>
326         </saml:Subject>
327       </saml:AuthenticationStatement>
328
329     </saml:Assertion>
330
331   </wsse:Security>
332 </S12:Header>
333 <S12:Body>
334   . . .
335 </S12:Body>
336 </S12:Envelope>

```

### 337 3.4 Identifying and Referencing Security Tokens

338 The **WSS: SOAP Message Security** specification defines the `<wsse:SecurityTokenReference>`  
339 element for referencing security tokens. Three forms of token references are defined by this element and  
340 the element schema includes provision for defining additional reference forms should they be necessary.  
341 The three forms of token references defined by the `<wsse:SecurityTokenReference>` element are  
342 defined as follows:

- 343 • A key identifier reference – a generic element (i.e., `<wsse:KeyIdentifier>`) that conveys a  
344 security token identifier as an `wsse:EncodedString` and indicates in its attributes (as necessary) the  
345 key identifier type (i.e., the `ValueType`), the identifier encoding type (i.e., the `EncodingType`), and  
346 perhaps other parameters used to reference the security token.

347 When a key identifier is used to reference a SAML assertion, it MUST contain as its element value the  
348 corresponding SAML assertion identifier. The key identifier MUST also contain a `ValueType`  
349 attribute and the value of this attribute MUST be the value from Table 2 corresponding to the version  
350 of the referenced assertion. The key identifier MUST NOT include an `EncodingType`<sup>4</sup> attribute and  
351 the element content of the key identifier MUST be encoded as `xs:string`.

352 When a key identifier is used to reference a V1.1 SAML assertion that is not contained in the same  
353 message as the key identifier, a `<saml:AuthorityBinding>` element MUST be contained in the

---

<sup>4</sup> "The Errata for Web Services Security: SOAP Message Security Version 1.0" (at <http://www.oasis-open.org/committees/wss>) removed the default designation from the `#Base64Binary` value for the `EncodingType` attribute of the `KeyIdentifier` element. Therefore, omitting a value for `EncodingType` and requiring that Base64 encoding not be performed, as specified by this profile, is consistent with the WS-Security Specification (including V1.1).

354 <wsse:SecurityTokenReference> element containing the key identifier. The contents of the  
355 <saml:AuthorityBinding> element MUST contain values sufficient for the intended recipients of  
356 the <wsse:SecurityTokenReference> to acquire the identified assertion from the intended  
357 Authority. To this end, the value of the AuthorityKind attribute of the  
358 <saml:AuthorityBinding> element MUST be "samlp:AssertionIdReference".

359 When a key Identifier is used to reference a SAML assertion contained in the same message as the  
360 key identifier, a <saml:AuthorityBinding> element MUST NOT be included in the  
361 <wsse:SecurityTokenReference> containing the key identifier.

362 A key identifier MUST NOT be used to reference a SAML V2.0 assertion if the assertion is NOT  
363 contained in the same message as the key identifier.

364 • A Direct or URI reference – a generic element (i.e., <wsse:Reference>) that identifies a security  
365 token by URI. If only a fragment identifier is specified, then the reference is to the security token within  
366 the document whose local identifier (e.g., wsu:Id attribute) matches the fragment identifier.  
367 Otherwise, the reference is to the (potentially external) security token identified by the URI.

368 A reference to a SAML V2.0 assertion that is NOT contained in the same message MUST be a Direct  
369 or URI reference. In this case, the value of the URI attribute must conform to the URI syntax defined in  
370 section 3.7.5.1 of [SAMLBindV2]. That is, an HTTP or HTTPS request with a single query string  
371 parameter named ID. The reference MUST also contain a wsse11:TokenType attribute and the  
372 value of this attribute MUST be the value from Table 3 identifying the assertion as a SAML V2.0  
373 security token. When a Direct reference is made to a SAML V2.0 Assertion, the Direct reference  
374 SHOULD NOT contain a ValueType attribute.

375 This profile does not describe the use of Direct or URI references to reference V1.1 SAML assertions.

376 • An Embedded reference – a reference that encapsulates a security token.

377 When an Embedded reference is used to encapsulate a SAML assertion, the SAML assertion MUST  
378 be included as a contained element within a <wsse:Embedded> element within a  
379 <wsse:SecurityTokenReference>.

380 This specification describes how SAML assertions may be referenced in four contexts:

381 • A SAML assertion may be referenced directly from a <wsse:Security> header element. In this  
382 case, the assertion is being conveyed by reference in the message.

383 • A SAML assertion may be referenced from a <ds:KeyInfo> element of a <ds:Signature>  
384 element in a <wsse:Security> header. In this case, the assertion contains a  
385 SubjectConfirmation element that identifies the key used in the signature calculation.

386 • A SAML assertion reference may be referenced from a <ds:Reference> element within the  
387 <ds:SignedInfo> element of a <ds:Signature> element in a <wsse:Security> header. In this  
388 case, the doubly-referenced assertion is signed by the containing signature.

389 • A SAML assertion reference may occur as encrypted content within an <xenc:EncryptedData>  
390 element referenced from a <xenc:DataReference> element within an <xenc:ReferenceList>  
391 element. In this case, the assertion reference (which may contain an embedded assertion) is  
392 encrypted.

393 In each of these contexts, the referenced assertion may be:

394 • local – in which case, it is included in the <wsse:Security> header containing the reference.

395 • remote – in which case it is not included in the <wsse:Security> header containing the reference,  
396 but may occur in another part of the SOAP message or may be available at the location identified by  
397 the reference which may be an assertion authority.

398 A SAML key identifier reference MUST be used for all (local and remote) references to SAML 1.1  
399 assertions. All (local and remote) references to SAML V2.0 assertions SHOULD be by Direct reference  
400 and all remote references to V2.0 assertions MUST be by Direct reference URI. A key identifier reference  
401 MAY be used to reference a local V2.0 assertion. To maintain compatibility with [Web Services Security:  
402 SOAP Message Security 1.0](#), the practice of referencing local SAML 1.1 assertions by Direct  
403 <wsse:SecurityTokenReference> reference is not defined by this profile.

404 Every key identifier, direct, or embedded reference to a SAML assertion SHOULD contain a  
 405 `wss11:TokenType` attribute and the value of this attribute MUST be the `value` from Table 3 that  
 406 identifies the type and version of the referenced security token. When the referenced assertion is a SAML  
 407 V2.0 Assertion the reference MUST contain a `wss11:TokenType` attribute (as described above).

Assertion Version	Value
V1.1	<a href="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0#SAMLAssertionID">http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0#SAMLAssertionID</a>
V2.0	<a href="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLID">http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLID</a>

408 Table-2 Key Identifier ValueType Attribute Values

Assertion Version	Value
V1.1	<a href="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV1.1">http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV1.1</a>
V2.0	<a href="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0">http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0</a>

409 Table-3 TokenType Attribute Values

410 The following subsections define the SAML assertion references that MUST be supported by conformant  
 411 implementations of this profile. A conformant implementation may choose to support the reference forms  
 412 corresponding to either or both V1.1 or V2.0 SAML assertions.

### 413 3.4.1 SAML Assertion Referenced from Header or Element

414 All conformant implementations MUST be able to process SAML assertion references occurring in a  
 415 `<wsse:Security>` header or in a header element other than a signature to acquire the corresponding  
 416 assertion. A conformant implementation MUST be able to process any such reference independent of the  
 417 confirmation method of the referenced assertion.

418 A SAML assertion may be referenced from a `<wsse:Security>` header or from an element (other than  
 419 a signature) in the header. The following example demonstrates the use of a key identifier in a  
 420 `<wsse:Security>` header to reference a local SAML V1.1 assertion.

```

421 <S12:Envelope>
422   <S12:Header>
423     <wsse:Security>
424       <saml:Assertion
425         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
426         IssueInstant="2003-04-17T00:46:02Z"
427         Issuer="www.opensaml.org"
428         MajorVersion="1"
429         MinorVersion="1"
430         . . .
431       </saml:Assertion>
432       <wsse:SecurityTokenReference wsu:Id="STR1"
433         wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
434 profile-1.1#SAMLV1.1">
435         <wsse:KeyIdentifier wsu:Id=""
436           ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
437 profile-1.0#SAMLAssertionID">
438           _a75adf55-01d7-40cc-929f-dbd8372ebdfc
439         </wsse:KeyIdentifier>
440       </wsse:SecurityTokenReference>
441     </wsse:Security>
442   </S12:Header>
443   <S12:Body>
444     . . .
  
```

```
445     </S12:Body>
446 </S12:Envelope>
```

447 The following example depicts the use of a key identifier reference to reference a local SAML V2.0  
448 assertion.

```
449 <wsse:SecurityTokenReference
450   wsu:Id="STR1"
451   wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
452   1.1#SAMLV2.0">
453   <wsse:KeyIdentifier wsu:Id="..."
454     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
455     1.1#SAMLID">
456     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
457   </wsse:KeyIdentifier>
458 </wsse:SecurityTokenReference>
```

459 A SAML V1.1 assertion that exists outside of a `<wsse:Security>` header may be referenced from the  
460 `<wsse:Security>` header element by including (in the `<wsse:SecurityTokenReference>`) a  
461 `<saml:AuthorityBinding>` element that defines the location, binding, and query that may be used to  
462 acquire the identified assertion at a SAML assertion authority or responder.

```
463 <wsse:SecurityTokenReference wsu:Id="STR1"
464   wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
465   profile-1.1#SAMLV1.1">
466   <saml:AuthorityBinding>
467     Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
468     Location="http://www.opensaml.org/SAML-Authority"
469     AuthorityKind="samlp:AssertionIdReference"
470   </saml:AuthorityBinding>
471   <wsse:KeyIdentifier
472     wsu:Id="..."
473     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
474     1.0#SAMLAssertionID">
475     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
476   </wsse:KeyIdentifier>
477 </wsse:SecurityTokenReference>
```

478 The following example depicts the use of a Direct or URI reference to reference a SAML V2.0 assertion  
479 that exists outside of a `<wsse:Security>` header.

```
480 </wsse:SecurityTokenReference
481   wsu:Id="..."
482   wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
483   profile-1.1#SAMLV2.0">
484   <wsse:Reference
485     wsu:Id="..."
486     URI="https://saml.example.edu/assertion-authority?ID=abcde">
487   </wsse:Reference>
488 </wsse:SecurityTokenReference>
```

### 489 3.4.2 SAML Assertion Referenced from KeyInfo

490 All conformant implementations MUST be able to process SAML assertion references occurring in the  
491 `<ds:KeyInfo>` element of a `<ds:Signature>` element in a `<wsse:Security>` header as defined by  
492 the holder-of-key confirmation method.

493 The following example depicts the use of a key identifier to reference a local V1.1 assertion from  
494 `<ds:KeyInfo>`.

```
495 <ds:KeyInfo>
496   <wsse:SecurityTokenReference wsu:Id="STR1"
497     wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
498     profile-1.1#SAMLV1.1">
499     <wsse:KeyIdentifier wsu:Id="..."
500       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
501       1.0#SAMLAssertionID">
```

```
502     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
503     </wsse:KeyIdentifier>
504     </wsse:SecurityTokenReference>
505 </ds:KeyInfo>
```

506 A local, V2.0 assertion may be referenced by replacing the values of the Key Identifier `ValueType` and  
507 reference `TokenType` attributes with the values defined in tables 2 and 3 (respectively) for SAML V2.0 as  
508 follows:

```
509 <ds:KeyInfo>
510   <wsse:SecurityTokenReference wsu:Id="STR1"
511     wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
512 profile-1.1#SAMLV2.0">
513     <wsse:KeyIdentifier wsu:Id="..."
514       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
515 1.1#SAMLID">
516       _a75adf55-01d7-40cc-929f-dbd8372ebdfc
517     </wsse:KeyIdentifier>
518   </wsse:SecurityTokenReference>
519 </ds:KeyInfo>
```

520 The following example demonstrates the use of a `<wsse:SecurityTokenReference>` containing a  
521 key identifier and a `<saml:AuthorityBinding>` to communicate information (location, binding, and  
522 query) sufficient to acquire the identified V1.1 assertion at an identified SAML assertion authority or  
523 responder.

```
524 <ds:KeyInfo>
525   <wsse:SecurityTokenReference wsu:Id="STR1"
526     wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
527 profile-1.1#SAMLV1.1">
528     <saml:AuthorityBinding>
529       Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
530       Location="http://www.opensaml.org/SAML-Authority"
531       AuthorityKind="samlp:AssertionIdReference"
532     </saml:AuthorityBinding>
533     <wsse:KeyIdentifier wsu:Id="..."
534       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
535 1.0#SAMLAssertionID">
536       _a75adf55-01d7-40cc-929f-dbd8372ebdfc
537     </wsse:KeyIdentifier>
538   </wsse:SecurityTokenReference>
539 </ds:KeyInfo>
```

540 Remote references to V2.0 assertions are made by Direct reference URI. The following example depicts  
541 the use of a Direct reference URI to reference a remote V2.0 assertion from `<ds:KeyInfo>`.

```
542 <ds:KeyInfo>
543   <wsse:SecurityTokenReference
544     wsu:id="STR1"
545     wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
546 profile-1.1#SAMLV2.0">
547     <wsse:Reference
548       wsu:id="..."
549       URI="https://saml.example.edu/assertion-authority?ID=abcde">
550     </wsse:Reference>
551   </wsse:SecurityTokenReference>
552 </ds:KeyInfo>
```

553 `<ds:KeyInfo>` elements may also occur in `<xenc:EncryptedData>` and `<xenc:EncryptedKey>`  
554 elements where they serve to identify the encryption key. `<ds:KeyInfo>` elements may also occur in  
555 SAML SubjectConfirmation elements where they identify a key that MUST be demonstrated to  
556 confirm the subject of the corresponding statement(s).

557 Conformant implementations of this profile are NOT required to process SAML assertion references  
558 occurring within the `<ds:KeyInfo>` elements within `<xenc:EncryptedData>`,  
559 `<xenc:EncryptedKey>`, or SAML SubjectConfirmation elements.

### 560 3.4.3 SAML Assertion Referenced from SignedInfo

561 Independent of the confirmation method of the referenced assertion, all conformant implementations  
562 MUST be able to process SAML assertions referenced by <wsse:SecurityTokenReference> from  
563 <ds:Reference> elements within the <ds:SignedInfo> element of a <ds:Signature> element in a  
564 <wsse:Security> header. Embedded references may be digested directly, thus effectively digesting the  
565 encapsulated assertion. Other <wsse:SecurityTokenReference> forms must be dereferenced for  
566 the referenced assertion to be digested.

567 The core specification, [WSS: SOAP Message Security](#), defines the STR Dereference transform to cause  
568 the replacement (in the digest stream) of a <wsse:SecurityTokenReference> with the contents of  
569 the referenced token. To digest any SAML assertion that is referenced by a non-embedded  
570 <wsse:SecurityTokenReference>, the STR Dereference transform MUST be specified and applied  
571 in the processing of the <ds:Reference>. Conversely, the STR Dereference transform MUST NOT be  
572 specified or applied when the <wsse:SecurityTokenReference>, not the referenced assertion, is to  
573 be digested.

574 The following example demonstrates the use of the STR Dereference transform to dereference a  
575 reference to a SAML V1.1 Assertion (i.e., Security Token) such that the digest operation is performed on  
576 the security token not its reference.

```
577 <wsse:SecurityTokenReference wsu:Id="STR1"  
578   wsse1:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-  
579   profile-1.1#SAMLV1.1">  
580   <saml:AuthorityBinding>  
581     Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"  
582     Location="http://www.opensaml.org/SAML-Authority"  
583     AuthorityKind= "samlp:AssertionIdReference"  
584   </saml:AuthorityBinding>  
585   <wsse:KeyIdentifier wsu:Id="..."  
586     ValueType="http://docs.oasis-open.org/wss/oasis-2004XX-wss-saml-token-  
587   profile-1.0#SAMLAssertionID">  
588     _a75adf55-01d7-40cc-929f-dbd8372ebdfc  
589   </wsse:KeyIdentifier>  
590 </wsse:SecurityTokenReference>  
591   . . .  
592 <ds:SignedInfo>  
593   <ds:CanonicalizationMethod  
594     Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />  
595   <ds:SignatureMethod  
596     Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />  
597   <ds:Reference URI="#STR1">  
598     <Transforms>  
599       <ds:Transform  
600         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
601   soap-message-security-1.0#STR-Transform">  
602         <wsse:TransformationParameters>  
603           <ds:CanonicalizationMethod  
604             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />  
605         </wsse:TransformationParameters>  
606       </ds:Transform>  
607     </Transforms>  
608     <ds:DigestMethod  
609       Algorithm= "http://www.w3.org/2000/09/xmldsig#sha1" />  
610     <ds:DigestValue>...</ds:DigestValue>  
611   </ds:Reference>  
612 </ds:SignedInfo>
```

614 Note that the URI appearing in the <ds:Reference> element identifies the  
615 <wsse:SecurityTokenReference> element by its wsu:Id value. Also note that the STR Dereference  
616 transform MUST contain (in <wsse:TransformationParameters>) a  
617 <ds:CanonicalizationMethod> that defines the algorithm to be used to serialize the input node set  
618 (of the referenced assertion).

619 As depicted in the other examples of this section, this profile establishes  
620 <wsse:SecurityTokenReference> forms for referencing V1.1, local V2.0, and remote V2.0  
621 assertions.

### 622 3.4.4 SAML Assertion Referenced from Encrypted Data Reference

623 Independent of the confirmation method of the referenced assertion, all conformant implementations  
624 MUST be able to process SAML assertion references occurring as encrypted content within the  
625 <xenc:EncryptedData> elements referenced by Id from the <xenc:DataReference> elements of  
626 <xenc:ReferenceList> elements. An <xenc:ReferenceList> element may occur either as a top-  
627 level element in a <wsse:Security> header, or embedded within an <xenc:EncryptedKey>  
628 element. In either case, the <xenc:ReferenceList> identifies the encrypted content.

629 Such references are similar in format to the references that MAY appear in the <ds:Reference>  
630 element within <ds:SignedInfo>, except the STR Dereference transform does not apply. As shown in  
631 the following example, an encrypted <wsse:SecurityTokenReference> (which may contain an  
632 embedded assertion) is referenced from an <xenc:DataReference> by including the identifier of the  
633 <xenc:EncryptedData> element that contains the encrypted <wsse:SecurityTokenReference>  
634 in the <xenc:DataReference>.

```
635 <xenc:EncryptedData Id="EncryptedSTR1">  
636 <ds:KeyInfo>  
637 . . .  
638 </ds:KeyInfo>  
639 <xenc:CipherData>  
640 <xenc:CipherValue>...</xenc:CipherValue>  
641 </xenc:CipherData>  
642 /xenc:EncryptedData>  
643 <xenc:ReferenceList>  
644 <xenc:DataReference URI="#EncryptedSTR1"/>  
645 </xenc:ReferenceList>
```

### 646 3.4.5 SAML Version Support and Backward Compatibility

647 An implementation of this profile MUST satisfy all of its requirements with respect to either or both SAML  
648 V1.1 or SAML V2.0 Assertions. An implementation that satisfies the requirements of this profile with  
649 respect to SAML V1.1 assertions MUST be able to fully interoperate with any fully compatible  
650 implementation of version 1.0 of this profile.

651 An implementation that does not satisfy the requirements of this profile with respect to SAML V1.1 or  
652 SAML V2.0 assertions MUST reject a message containing a <wsse:Security> header that references  
653 or conveys an assertion of the unsupported version. When a message containing an unsupported  
654 assertion version is detected, the receiver MAY choose to respond with an appropriate fault as defined in  
655 Section 3.6, "Error Codes".

### 656 3.5 Subject Confirmation of SAML Assertions

657 The SAML profile of [WSS: SOAP Message Security](#) requires that systems support the holder-of-key and  
658 sender-vouches methods of subject confirmation. It is strongly RECOMMENDED that an XML signature  
659 be used to establish the relationship between the message and the statements of the attached assertions.  
660 This is especially RECOMMENDED whenever the SOAP message exchange is conducted over an  
661 unprotected transport.

662 Any processor of SAML assertions MUST conform to the required validation and processing rules defined  
663 in the corresponding SAML specification including the validation of assertion signatures, the processing of  
664 <saml:Condition> elements within assertions, and the processing of  
665 <saml2:SubjectConfirmationData> attributes. [\[SAMLCoreV1\]](#) defines the validation and  
666 processing rules for V1.1 assertions, while [\[SAMLCoreV2\]](#) is authoritative for V2.0 assertions.

667 The following table enumerates the mandatory subject confirmation methods and summarizes their  
668 associated processing models:

Mechanism	RECOMMENDED Processing Rules
Urn:oasis:names:tc:SAML:1.0:cm:holder-of-key Or urn:oasis:names:tc:SAML:2.0:cm:holder-of-key	The attesting entity demonstrates knowledge of a confirmation key identified in a holder-of-key <code>SubjectConfirmation</code> element within the assertion.
Urn:oasis:names:tc:SAML:1.0:cm:sender-vouches Or urn:oasis:names:tc:SAML:2.0:cm:sender-vouches	The attesting entity, (presumed to be) different from the subject, vouches for the verification of the subject. The receiver <b>MUST</b> have an existing trust relationship with the attesting entity. The attesting entity <b>MUST</b> protect the assertion in combination with the message content against modification by another party. See also section 4.

669 Note that the high level processing model described in the following sections does not differentiate  
 670 between the attesting entity and the message sender as would be necessary to guard against replay  
 671 attacks. The high-level processing model also does not take into account requirements for authentication  
 672 of receiver by sender, or for message or assertion confidentiality. These concerns must be addressed by  
 673 means other than those described in the high-level processing model (i.e., section 3.1).

### 674 3.5.1 Holder-of-key Subject Confirmation Method

675 The following sections describe the holder-of-key method of establishing the correspondence between a  
 676 SOAP message and the subject and claims of SAML assertions added to the SOAP message according  
 677 to this specification.

#### 678 3.5.1.1 Attesting Entity

679 An attesting entity demonstrates that it is authorized to act as the subject of a holder-of-key confirmed  
 680 SAML statement by demonstrating knowledge of any key identified in a holder-of-key  
 681 `SubjectConfirmation` element associated with the statement by the assertion containing the  
 682 statement. Statements attested for by the holder-of-key method **MUST** be associated, within their  
 683 containing assertion, with one or more holder-of-key `SubjectConfirmation` elements.

684 The `SubjectConfirmation` elements **MUST** include a `<ds:KeyInfo>` element that identifies a public  
 685 or secret key<sup>5</sup> that can be used to confirm the identity of the subject.

686 To satisfy the associated confirmation method processing to be performed by the message receiver, the  
 687 attesting entity **MUST** demonstrate knowledge of the confirmation key. The attesting entity **MAY**  
 688 accomplish this by using the confirmation key to sign content within the message and by including the  
 689 resulting `<ds:Signature>` element in the `<wsse:Security>` header. `<ds:Signature>` elements  
 690 produced for this purpose **MUST** conform to the canonicalization and token pre-pending rules defined in  
 691 the [WSS: SOAP Message Security](#) specification.

---

<sup>5</sup>[SAMLCoreV1] defines `KeyInfo` of `SubjectConfirmation` as containing a “cryptographic key held by the subject”. Demonstration of this key is sufficient to establish who is (or may act as the) subject. Moreover, since it cannot be proven that a confirmation key is known (or known only) by the subject whose identity it establishes, requiring that the key be held by the subject is an untestable requirement that adds nothing to the strength of the confirmation mechanism. In [SAMLCoreV2], the OASIS Security Services Technical Committee agreed to remove the phrase “held by the subject” from the definition of `KeyInfo` within `SubjectConfirmation(Data)`.

692 SAML assertions that contain a holder-of-key `SubjectConfirmation` element SHOULD contain a  
693 `<ds:Signature>` element that protects the integrity of the confirmation `<ds:KeyInfo>` established by  
694 the assertion authority.

695 The canonicalization method used to produce the `<ds:Signature>` elements used to protect the  
696 integrity of SAML assertions MUST support the validation of these `<ds:Signature>` elements in  
697 contexts (such as `<wsse:Security>` header elements) other than those in which the signatures were  
698 calculated.

### 699 3.5.1.2 Receiver

700 Of the SAML assertions it selects for processing, a message receiver MUST NOT accept statements of  
701 these assertions based on a holder-of-key `SubjectConfirmation` element defined for the statements  
702 (within the assertion) unless the receiver has validated the integrity of the assertion and the attesting entity  
703 has demonstrated knowledge of a key identified within the confirmation element.

704 If the receiver determines that the attesting entity has demonstrated knowledge of a subject confirmation  
705 key, then the subjects and claims of the SAML statements confirmed by the key MAY be attributed to the  
706 attesting entity and any content of the message whose integrity is protected by the key MAY be  
707 considered to have been provided by the attesting entity.

### 708 3.5.1.3 Example V1.1

709 The following example illustrates the use of the holder-of-key subject confirmation method to establish the  
710 correspondence between the SOAP message and the subject of statements of the SAML V1.1 assertions  
711 in the `<wsse:Security>` header:

```
712 <?xml version="1.0" encoding="UTF-8"?>
713 <S12:Envelope>
714   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
715   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
716   <S12:Header>
717
718     <wsse:Security>
719       <saml:Assertion
720         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
721         IssueInstant="2005-05-27T16:53:33.173Z"
722         Issuer="www.opensaml.org"
723         MajorVersion="1"
724         MinorVersion="1"
725         xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
726         <saml:Conditions>
727           NotBefore="2005-05-27T16:53:33.173Z"
728           NotOnOrAfter="2005-05-27T16:58:33.17302Z"/>
729         <saml:AttributeStatement>
730           <saml:Subject>
731             <saml:NameIdentifier
732               NameQualifier="www.example.com"
733               Format="urn:oasis:names:tc:SAML:1.1:nameid-
734 format:X509SubjectName">
735               uid=joe,ou=people,ou=saml-demo,o=baltimore.com
736             </saml:NameIdentifier>
737             <saml:SubjectConfirmation>
738               <saml:ConfirmationMethod>
739                 urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
740               </saml:ConfirmationMethod>
741               <ds:KeyInfo>
742                 <ds:KeyValue>...</ds:KeyValue>
743               </ds:KeyInfo>
744             </saml:SubjectConfirmation>
745           </saml:Subject>
746           <saml:Attribute
747             AttributeName="MemberLevel"
```

```

748         AttributeNamespace="http://www.oasis-
749 open.org/Catalyst2002/attributes">
750         <saml:AttributeValue>gold</saml:AttributeValue>
751     </saml:Attribute>
752     <saml:Attribute
753         AttributeName="E-mail"
754         AttributeNamespace="http://www.oasis-
755 open.org/Catalyst2002/attributes">
756         <saml:AttributeValue>joe@yahoo.com</saml:AttributeValue>
757     </saml:Attribute>
758 </saml:AttributeStatement>
759 <ds:Signature>...</ds:Signature>
760 </saml:Assertion>
761
762 <ds:Signature>
763     <ds:SignedInfo>
764         <ds:CanonicalizationMethod
765             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
766         <ds:SignatureMethod
767             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
768         <ds:Reference
769             URI="#MsgBody">
770             <ds:DigestMethod
771                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
772             <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
773         </ds:Reference>
774     </ds:SignedInfo>
775     <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
776     <ds:KeyInfo>
777         <wsse:SecurityTokenReference wsu:Id="STR1"
778             wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
779 token-profile-1.1#SAMLV1.1">
780             <wsse:KeyIdentifier wsu:Id="..."
781                 ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
782 profile-1.0#SAMLAssertionID">
783                 _a75adf55-01d7-40cc-929f-dbd8372ebdfc
784             </wsse:KeyIdentifier>
785         </wsse:SecurityTokenReference>
786     </ds:KeyInfo>
787 </ds:Signature>
788 </wsse:Security>
789 </S12:Header>
790
791 <S12:Body wsu:Id="MsgBody">
792     <ReportRequest>
793         <TickerSymbol>SUNW</TickerSymbol>
794     </ReportRequest>
795 </S12:Body>
796 </S12:Envelope>

```

### 797 3.5.1.4 Example V2.0

798 The following example illustrates the use of the holder-of-key subject confirmation method to establish the  
799 correspondence between the SOAP message and the subject of the SAML V2.0 assertion in the

800 <wsse:Security> header:

```

801 <?xml version="1.0" encoding="UTF-8"?>
802 <S12:Envelope>
803     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
804     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
805     <S12:Header>
806
807         <wsse:Security>
808             <saml2:Assertion
809                 ...

```

```

810         ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
811         ...>
812     <saml2:Subject>
813     <saml2:NameID>
814         ...
815     </saml2:NameID>
816     <saml2:SubjectConfirmation
817         Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
818         <saml2:SubjectConfirmationData
819             xsi:type="saml2:KeyInfoConfirmationDataType">
820             <ds:KeyInfo>
821                 <ds:KeyValue>...</ds:KeyValue>
822             </ds:KeyInfo>
823         </saml2:SubjectConfirmationData>
824     </saml2:SubjectConfirmation>
825 </saml2:Subject>
826 <saml2:Statement>
827     ...
828 </saml2:Statement>
829     <ds:Signature>...</ds:Signature>
830 </saml2:Assertion>
831
832 <ds:Signature>
833     <ds:SignedInfo>
834         <ds:CanonicalizationMethod
835             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
836         <ds:SignatureMethod
837             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
838         <ds:Reference
839             URI="#MsgBody">
840             <ds:DigestMethod
841                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
842             <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
843         </ds:Reference>
844     </ds:SignedInfo>
845     <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
846     <ds:KeyInfo>
847         <wsse:SecurityTokenReference wsu:Id="STR1"
848             wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
849 token-profile-1.1#SAMLV2.0">
850             <wsse:KeyIdentifier wsu:Id="..."
851                 ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
852 profile-1.1#SAMLID">
853                 _a75adf55-01d7-40cc-929f-dbd8372ebdfc
854             </wsse:KeyIdentifier>
855         </wsse:SecurityTokenReference>
856     </ds:KeyInfo>
857 </ds:Signature>
858 </wsse:Security>
859 </S12:Header>
860
861 <S12:Body wsu:Id="MsgBody">
862     <ReportRequest>
863         <TickerSymbol>SUNW</TickerSymbol>
864     </ReportRequest>
865 </S12:Body>
866 </S12:Envelope>

```

### 867 3.5.2 Sender-vouches Subject Confirmation Method

868 The following sections describe the sender-vouches method of establishing the correspondence between  
869 a SOAP message and the SAML assertions added to the SOAP message according to the SAML profile  
870 of [WSS: SOAP Message Security](#).

### 871 3.5.2.1 Attesting Entity

872 An attesting entity uses the sender-vouches confirmation method to assert that it is acting on behalf of the  
873 subject of SAML statements attributed with a sender-vouches `SubjectConfirmation` element.  
874 Statements attested for by the sender-vouches method MUST be associated, within their containing  
875 assertion, with one or more sender-vouches `SubjectConfirmation` elements.

876 To satisfy the associated confirmation method processing of the receiver, the attesting entity MUST  
877 protect the vouched for SOAP message content such that the receiver can determine when it has been  
878 altered by another party. The attesting entity MUST also cause the vouched for statements (as necessary)  
879 and their binding to the message contents to be protected such that unauthorized modification can be  
880 detected. The attesting entity MAY satisfy these requirements by including in the corresponding  
881 `<wsse:Security>` header a `<ds:Signature>` element that it prepares by using its key to sign the  
882 relevant message content and assertions. As defined by the [XML Signature](#) specification, the attesting  
883 entity MAY identify its key by including a `<ds:KeyInfo>` element within the `<ds:Signature>` element.

884 A `<ds:Signature>` element produced for this purpose MUST conform to the canonicalization and  
885 token pre-pending rules defined in the [WSS: SOAP Message Security](#) specification.

### 886 3.5.2.2 Receiver

887 Of the SAML assertions it selects for processing, a message receiver MUST NOT accept statements of  
888 these assertions based on a sender-vouches `SubjectConfirmation` element defined for the  
889 statements (within the assertion) unless the assertions and SOAP message content being vouched for are  
890 protected (as described above) by an attesting entity who is trusted by the receiver to act as the subjects  
891 and with the claims of the statements.

### 892 3.5.2.3 Example V1.1

893 The following example illustrates an attesting entity's use of the sender-vouches subject confirmation  
894 method with an associated `<ds:Signature>` element to establish its identity and to assert that it has  
895 sent the message body on behalf of the subject(s) of the V1.1 assertion referenced by "STR1".

896 The assertion referenced by "STR1" is not included in the message. "STR1" is referenced by  
897 `<ds:Reference>` from `<ds:SignedInfo>`. The `ds:Reference` includes the STR-transform to  
898 cause the assertion, not the `<SecurityTokenReference>` to be included in the digest calculation.  
899 "STR1" includes a `<saml:AuthorityBinding>` element that utilizes the remote assertion referencing  
900 technique depicted in the example of section 3.3.3.

901 The SAML V1.1 assertion embedded in the header and referenced by "STR2" from `<ds:KeyInfo>`  
902 corresponds to the attesting entity. The private key corresponding to the public confirmation key occurring  
903 in the assertion is used to sign together the message body and assertion referenced by "STR1".

```
904 <?xml version="1.0" encoding="UTF-8"?>
905 <S12:Envelope>
906   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
907   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
908   <S12:Header>
909     <wsse:Security>
910
911     <saml:Assertion
912       AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
913       IssueInstant="2005-05-27T16:53:33.173Z"
914       Issuer="www.opensaml.org"
915       MajorVersion="1"
916       MinorVersion="1"
917       xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
918     <saml:Conditions>
919       NotBefore="2005-05-27T16:53:33.173Z"
920       NotOnOrAfter="2005-05-27T16:58:33.173Z"/>
921     <saml:AttributeStatement>
922       <saml:Subject>
923         <saml:NameIdentifier
```

```

924         NameQualifier="www.example.com"
925         Format="urn:oasis:names:tc:SAML:1.1:nameid-
926 format:X509SubjectName">
927         uid=proxy,ou=system,ou=saml-demo,o=baltimore.com
928     </saml:NameIdentifier>
929     <saml:SubjectConfirmation>
930         <saml:ConfirmationMethod>
931             urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
932         </saml:ConfirmationMethod>
933         <ds:KeyInfo>
934             <ds:KeyValue>...</ds:KeyValue>
935         </ds:KeyInfo>
936     </saml:SubjectConfirmation>
937 </saml:Subject>
938 <saml:Attribute
939     . . .
940 </saml:Attribute>
941     . . .
942 </saml:AttributeStatement>
943 </saml:Assertion>
944
945     <wsse:SecurityTokenReference wsu:Id="STR1">
946         wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
947 profile-1.1#SAMLV1.1">
948         <saml:AuthorityBinding>
949             Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
950             Location="http://www.opensaml.org/SAML-Authority"
951             AuthorityKind="samlp:AssertionIdReference"
952         </saml:AuthorityBinding>
953         <wsse:KeyIdentifier wsu:Id="..."
954             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
955 profile-1.0#SAMLAssertionID">
956             _a75adf55-01d7-40cc-929f-dbd8372ebdbe
957         </wsse:KeyIdentifier>
958     </wsse:SecurityTokenReference>
959
960     <ds:Signature>
961         <ds:SignedInfo>
962             <ds:CanonicalizationMethod
963                 Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
964             <ds:SignatureMethod
965                 Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
966             <ds:Reference URI="#STR1">
967                 <Transforms>
968                     <ds:Transform
969                         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-
970 200401-wss-soap-message-security-1.0#STR-Transform">
971                         <wsse:TransformationParameters>
972                             <ds:CanonicalizationMethod
973                                 Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
974                             </wsse:TransformationParameters>
975                         </ds:Transform>
976                     </Transforms>
977                     <ds:DigestMethod
978                         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
979                     <ds:DigestValue>...</ds:DigestValue>
980                 </ds:Reference>
981             <ds:Reference URI="#MsgBody">
982                 <ds:DigestMethod
983                     Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
984                 <ds:DigestValue>...</ds:DigestValue>
985             </ds:Reference>
986         </ds:SignedInfo>
987         <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
988         <ds:KeyInfo>
989             <wsse:SecurityTokenReference wsu:Id="STR2"

```

```

990         wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
991 token-profile-1.1#SAMLV1.1">
992         <wsse:KeyIdentifier wsu:Id="..."
993           ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
994 profile-1.0#SAMLAssertionID">
995           _a75adf55-01d7-40cc-929f-dbd8372ebdfc
996         </wsse:KeyIdentifier>
997         </wsse:SecurityTokenReference>
998       </ds:KeyInfo>
999     </ds:Signature>
1000   </wsse:Security>
1001 </S12:Header>
1002
1003   <S12:Body wsu:Id="MsgBody">
1004     <ReportRequest>
1005       <TickerSymbol>SUNW</TickerSymbol>
1006     </ReportRequest>
1007   </S12:Body>
1008 </S12:Envelope>

```

### 1009 3.5.2.4 Example V2.0

1010 The following example illustrates the mapping of the preceding example to SAML V2.0 assertions.

```

1011 <?xml version="1.0" encoding="UTF-8"?>
1012 <S12:Envelope>
1013   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1014   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
1015   <S12:Header>
1016
1017     <wsse:Security>
1018       <saml2:Assertion
1019         ...
1020         ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
1021         ...>
1022         <saml2:Subject>
1023           <saml2:NameID>
1024             ...
1025           </saml2:NameID>
1026           <saml2:SubjectConfirmation
1027             Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
1028             <saml2:SubjectConfirmationData
1029               xsi:type="saml2:KeyInfoConfirmationDataType">
1030               <ds:KeyInfo>
1031                 <ds:KeyValue>...</ds:KeyValue>
1032               </ds:KeyInfo>
1033             </saml2:SubjectConfirmationData>
1034           </saml2:SubjectConfirmation>
1035         </saml2:Subject>
1036         <saml2:Statement>
1037           ...
1038         </saml2:Statement>
1039         <ds:Signature>...</ds:Signature>
1040       </saml2:Assertion>
1041
1042       <wsse:SecurityTokenReference wsu:Id="STR1"
1043         wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1044 profile-1.1#SAMLV2.0">
1045         <wsse:Reference wsu:Id="..."
1046           URI="https://www.opensaml.org?_a75adf55-01d7-40cc-929f-
1047 dbd8372ebdbe">
1048         </wsse:Reference>
1049       </wsse:SecurityTokenReference>
1050
1051       <ds:Signature>
1052         <ds:SignedInfo>

```

```

1053     <ds:CanonicalizationMethod
1054       Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1055     <ds:SignatureMethod
1056       Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
1057     <ds:Reference URI="#STR1">
1058       <Transforms>
1059         <ds:Transform
1060           Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
1061 wss-soap-message-security-1.0#STR-Transform">
1062           <wsse:TransformationParameters>
1063             <ds:CanonicalizationMethod
1064               Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1065             </wsse:TransformationParameters>
1066           </ds:Transform>
1067         </Transforms>
1068       </ds:Reference>
1069     <ds:DigestMethod
1070       Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1071     <ds:DigestValue>...</ds:DigestValue>
1072   </ds:Reference>
1073 <ds:Reference URI="#MsgBody">
1074   <ds:DigestMethod
1075     Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1076   <ds:DigestValue>...</ds:DigestValue>
1077 </ds:Reference>
1078 </ds:SignedInfo>
1079 <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
1080 <ds:KeyInfo>
1081   <wsse:SecurityTokenReference wsu:Id="STR2"
1082     wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
1083 token-profile-1.1#SAMLV2.0">
1084     <wsse:KeyIdentifier wsu:Id="..."
1085       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1086 profile-1.1#SAMLID">
1087       _a75adf55-01d7-40cc-929f-dbd8372ebdfc
1088     </wsse:KeyIdentifier>
1089   </wsse:SecurityTokenReference>
1090 </ds:KeyInfo>
1091 </ds:Signature>
1092 </wsse:Security>
1093 </S12:Header>
1094
1095 <S12:Body wsu:Id="MsgBody">
1096   <ReportRequest>
1097     <TickerSymbol>SUNW</TickerSymbol>
1098   </ReportRequest>
1099 </S12:Body>
1100 </S12:Envelope>

```

### 1101 3.5.3 Bearer Confirmation Method

1102 This profile does NOT require message receivers to establish the relationship between a received  
1103 message and the statements of any bearer confirmed (i.e., confirmation method  
1104 urn:oasis:names:tc:SAML:1.0:cm:bearer) assertions conveyed or referenced from the message.  
1105 Conformant implementations of this profile MUST be able to process references and convey bearer  
1106 assertions within <wsse:Security> headers. Any additional processing requirements that pertain  
1107 specifically to bearer confirmed assertions are outside the scope of this profile.

### 1108 3.6 Error Codes

1109 When a system that implements the SAML token profile of [WSS: SOAP Message Security](#) does not  
1110 perform its normal processing because of an error detected during the processing of a security header, it  
1111 MAY choose to report the cause of the error using the SOAP fault mechanism. The SAML token profile of

1112 [WSS: SOAP Message Security](#) does not require that SOAP faults be returned for such errors, and  
 1113 systems that choose to return faults SHOULD take care not to introduce any security vulnerabilities as a  
 1114 result of the information returned in error responses.

1115 Systems that choose to return faults SHOULD respond with the error codes and fault strings defined in the  
 1116 [WSS: SOAP Message Security](#) specification. The RECOMMENDED correspondence between the  
 1117 common assertion processing failures and the error codes defined in [WSS: SOAP Message Security](#) are  
 1118 defined in the following table:

Assertion Processing Error	RECOMMENDED Error(Faultcode)
A referenced SAML assertion could not be retrieved.	wsse:SecurityTokenUnavailable
An assertion contains a <saml:Condition> element that the receiver does not understand.	wsse:UnsupportedSecurityToken
A signature within an assertion or referencing an assertion is invalid.	wsse:FailedCheck
The issuer of an assertion is not acceptable to the receiver.	wsse:InvalidSecurityToken
The receiver does not understand the extension schema used in an assertion.	wsse:UnsupportedSecurityToken
The receiver does not support the SAML version of a referenced or included assertion.	wsse:UnsupportedSecurityToken

1119 The preceding table defines fault codes in a form suitable for use with SOAP 1.1. The [WSS: SOAP](#)  
 1120 [Message Security](#) specification describes how to map SOAP 1.1 fault constructs to the SOAP 1.2 fault  
 1121 constructs.

---

## 1122 4 Threat Model and Countermeasures (non- 1123 normative)

1124 This document defines the mechanisms and procedures for securely attaching SAML assertions to SOAP  
1125 messages. SOAP messages are used in multiple contexts, specifically including cases where the  
1126 message is transported without an active session, the message is persisted, or the message is routed  
1127 through a number of intermediaries. Such a general context of use suggests that users of this profile must  
1128 be concerned with a variety of threats.

1129 In general, the use of SAML assertions with [WSS: SOAP Message Security](#) introduces no new threats  
1130 beyond those identified for SAML or by the [WSS: SOAP Message Security](#) specification. The following  
1131 sections provide an overview of the characteristics of the threat model, and the countermeasures that  
1132 SHOULD be adopted for each perceived threat.

### 1133 4.1 Eavesdropping

1134 Eavesdropping is a threat to the SAML token profile of [WSS: SOAP Message Security](#) in the same  
1135 manner as it is a threat to any network protocol. The routing of SOAP messages through intermediaries  
1136 increases the potential incidences of eavesdropping. Additional opportunities for eavesdropping exist  
1137 when SOAP messages are persisted.

1138 To provide maximum protection from eavesdropping, assertions, assertion references, and sensitive  
1139 message content SHOULD be encrypted such that only the intended audiences can view their content.  
1140 This approach removes threats of eavesdropping in transit, but MAY not remove risks associated with  
1141 storage or poor handling by the receiver.

1142 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or  
1143 references from eavesdropping while in transport, but message content MUST be encrypted above the  
1144 transport if it is to be protected from eavesdropping by intermediaries.

### 1145 4.2 Replay

1146 Reliance on authority-protected (e.g., signed) assertions with a holder-of-key subject confirmation  
1147 mechanism precludes all but a holder of the key from binding the assertions to a SOAP message.  
1148 Although this mechanism effectively restricts data origin to a holder of the confirmation key, it does not, by  
1149 itself, provide the means to detect the capture and resubmission of the message by other parties.

1150 Assertions that contain a sender-vouches confirmation mechanism introduce another dimension to replay  
1151 vulnerability if the assertions impose no restriction on the entities that may use or reuse the assertions.

1152 Replay attacks can be detected by receivers if message senders include additional message identifying  
1153 information (e.g., timestamps, nonces, and or recipient identifiers) within origin-protected message  
1154 content and receivers check this information against previously received values.

### 1155 4.3 Message Insertion

1156 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message insertion attacks.

### 1157 4.4 Message Deletion

1158 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message deletion attacks.

### 1159 4.5 Message Modification

1160 Messages constructed according to this specification are protected from message modification if receivers  
1161 can detect unauthorized modification of relevant message content. Therefore, it is strongly  
1162 RECOMMENDED that all relevant and immutable message content be signed by an attesting entity.  
1163 Receivers SHOULD only consider the correspondence between the subject of the SAML assertions and

1164 the SOAP message content to have been established for those portions of the message that are protected  
1165 by the attesting entity against modification by another entity.

1166 To ensure that message receivers can have confidence that received assertions have not been forged or  
1167 altered since their issuance, SAML assertions appearing in or referenced from `<wsse:Security>`  
1168 header elements MUST be protected against unauthorized modification (e.g., signed) by their issuing  
1169 authority or the attesting entity (as the case warrants). It is strongly RECOMMENDED that an attesting  
1170 entity sign any `<saml:Assertion>` elements that it is attesting for and that are not signed by their  
1171 issuing authority.

1172 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or  
1173 assertion references from modification while in transport, but signatures are required to extend such  
1174 protection through intermediaries.

## 1175 **4.6 Man-in-the-Middle**

1176 Assertions with a holder-of-key subject confirmation method are not vulnerable to a MITM attack.  
1177 Assertions with a sender-vouches subject confirmation method are vulnerable to MITM attacks to the  
1178 degree that the receiver does not have a trusted binding of key to the attesting entity's identity.

---

## 5 References

- 1179
- 1180 **[GLOSSARY]** Informational RFC 2828, "[Internet Security Glossary](#)," May 2000.
- 1181 **[KEYWORDS]** S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," [RFC 2119](#), Harvard University, March 1997
- 1182
- 1183 **[SAMLBindV1]** Oasis Standard, E. Maler, P.Mishra, and R. Philpott (Editors), [Bindings and Profiles for the OASIS Security Assertion Markup Language \(SAML\) V1.1](#), September 2003.
- 1184
- 1185
- 1186 **[SAMLBindV2]** Oasis Standard, S. Cantor, F. Hirsch, J. Kemp, R. Philpott, E. Maler (Editors), [Bindings for the OASIS Security Assertion Markup Language \(SAML\) V2.0](#), March 2005.
- 1187
- 1188
- 1189 **[SAMLCoreV1]** Oasis Standard, E. Maler, P.Mishra, and R. Philpott (Editors), [Assertions and Protocols for the OASIS Security Assertion Markup Language \(SAML\) V1.1](#), September 2003.
- 1190
- 1191
- 1192 **[SAMLCoreV2]** Oasis Standard, S. Cantor, J. Kemp, R. Philpott, E. Maler (Editors), [Assertions and Protocol for the OASIS Security Assertion Markup Language \(SAML\) V2.0](#), March 2005.
- 1193
- 1194
- 1195 **[SOAP]** W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.
- 1196 W3C Working Draft, Nilo Mitra (Editor), [SOAP Version 1.2 Part 0: Primer](#), June 2002.
- 1197
- 1198 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 1: Messaging Framework](#), June 2002.
- 1199
- 1200
- 1201 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 2: Adjuncts](#), June 2002.
- 1202
- 1203
- 1204 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," [RFC 2396](#), MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998.
- 1205
- 1206
- 1207 **[WS-SAML]** Contribution to the WSS TC, P. Mishra (Editor), [WS-Security Profile of the Security Assertion Markup Language \(SAML\) Working Draft 04](#), Sept 2002.
- 1208
- 1209 **[WSS: SAML Token Profile]** Oasis Standard, P. Hallem-Baker, A. Nadalin, C. Kaler, R. Monzillo (Editors), [Web Services Security: SAML Token Profile 1.0](#), December 2004.
- 1210
- 1211 **[WSS: SOAP Message Security V1.0]** Oasis Standard, A. Nadalin, C.Kaler, P. Hallem-Baker, R. Monzillo (Editors), [Web Services Security: SOAP Message Security 1.0 \(WS-Security 2004\)](#), August 2003.
- 1212
- 1213
- 1214 **[WSS: SOAP Message Security]** Oasis Standard, A. Nadalin, C.Kaler, R. Monzillo, P. Hallem-Baker, (Editors), [Web Services Security: SOAP Message Security 1.1 \(WS-Security 2004\)](#), December 2005.
- 1215
- 1216
- 1217 **[XML-ns]** W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.
- 1218 **[XML Signature]** W3C Recommendation, "[XML Signature Syntax and Processing](#)," 12 February 2002.
- 1219
- 1220 **[XML Token]** Contribution to the WSS TC, Chris Kaler (Editor),
- 1221 WS-Security Profile for XML-based Tokens, August 2002.

## Appendix A. Acknowledgements

### Current Contributors:

Michael	Hu	Actional
Maneesh	Sahu	Actional
Duane	Nickull	Adobe Systems
Gene	Thurston	AmberPoint
Frank	Siebenlist	Argonne National Laboratory
Hal	Lockhart	BEA Systems
Denis	Pilipchuk	BEA Systems
Corinna	Witt	BEA Systems
Steve	Anderson	BMC Software
Rich	Levinson	Computer Associates
Thomas	DeMartini	ContentGuard
Merlin	Hughes	Cybertrust
Dale	Moberg	Cyclone Commerce
Rich	Salz	Datapower
Sam	Wei	EMC
Dana S.	Kaufman	Forum Systems
Toshihiro	Nishimura	Fujitsu
Kefeng	Chen	GeoTrust
Irving	Reid	Hewlett-Packard
Kojiro	Nakayama	Hitachi
Paula	Austel	IBM
Derek	Fu	IBM
Maryann	Hondo	IBM
Kelvin	Lawrence	IBM
Michael	McIntosh	IBM
Anthony	Nadalin	IBM
Nataraj	Nagaratnam	IBM
Bruce	Rich	IBM
Ron	Williams	IBM
Don	Flinn	Individual
Kate	Cherry	Lockheed Martin
Paul	Cotton	Microsoft
Vijay	Gajjala	Microsoft
Martin	Gudgin	Microsoft
Chris	Kaler	Microsoft
Frederick	Hirsch	Nokia
Abbie	Barbir	Nortel
Prateek	Mishra	Oracle
Vamsi	Motukuru	Oracle
Ramana	Turlapi	Oracle
Ben	Hammond	RSA Security
Rob	Philpott	RSA Security
Blake	Dournaee	Sarvega
Sundeeep	Peechu	Sarvega
Coumara	Radja	Sarvega
Pete	Wenzel	SeeBeyond
Manveen	Kaur	Sun Microsystems
Ronald	Monzillo	Sun Microsystems
Jan	Alexander	Systinet
Symon	Chang	TIBCO Software
John	Weiland	US Navy
Hans	Granqvist	VeriSign

Phillip	Hallem-Baker	VeriSign
Hemma	Prafullchandra	VeriSign

**Previous Contributors:**

Peter	Dapkus	BEA
Guillermo	Lao	ContentGuard
TJ	Pannu	ContentGuard
Xin	Wang	ContentGuard
Shawn	Sharp	Cyclone Commerce
Ganesh	Vaideeswaran	Documentum
Tim	Moses	Entrust
Carolina	Canales-Valenzuela	Ericsson
Tom	Rutt	Fujitsu
Yutaka	Kudo	Hitachi
Jason	Rouault	HP
Bob	Blakley	IBM
Joel	Farrell	IBM
Satoshi	Hada	IBM
Hiroshi	Maruyama	IBM
David	Melgar	IBM
Kent	Tamura	IBM
Wayne	Vicknair	IBM
Phil	Griffin	Individual
Mark	Hayes	Individual
John	Hughes	Individual
Peter	Rostin	Individual
Davanum	Srinivas	Individual
Bob	Morgan	Individual/Internet2
Bob	Atkinson	Microsoft
Keith	Ballinger	Microsoft
Allen	Brown	Microsoft
Giovanni	Della-Libera	Microsoft
Alan	Geller	Microsoft
Johannes	Klein	Microsoft
Scott	Konersmann	Microsoft
Chris	Kurt	Microsoft
Brian	LaMacchia	Microsoft
Paul	Leach	Microsoft
John	Manferdelli	Microsoft
John	Shewchuk	Microsoft
Dan	Simon	Microsoft
Hervey	Wilson	Microsoft
Jeff	Hodges	Neustar
Senthil	Sengodan	Nokia
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Vipin	Samar	Oracle
Jerry	Schwarz	Oracle
Eric	Gravengaard	Reactivity
Andrew	Nash	Reactivity
Stuart	King	Reed Elsevier
Martijn	de Boer	SAP
Jonathan	Tourzan	Sony
Yassir	Elley	Sun
Michael	Nguyen	The IDA of Singapore
Don	Adams	TIBCO

Morten	Jorgensen	Vordel
--------	-----------	--------