



Web Services Security: SAML Token Profile 1.1

**OASIS Committee Draft Standard, 12
SeptemberFebruary 2006**

Document Identifier:

wss-v1.1-spec-~~errata-cdes~~-SAMLTokenProfile

OASIS Identifier:

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

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 is an **OASIS Standard Committee Draft incorporating committee approved errata to the OASIS Standard.** ~~The standard~~ was approved by the OASIS membership on 1 February 2006. ~~Check the location noted below for possible errata to this specification.~~

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.

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).

The non-normative errata for this specification is located at 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 2002-2006. 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.

65 Table of Contents

66	1Introduction.....	5
67	1.1Goals.....	5
68	1.1.1Non-Goals.....	5
69	2Notations and Terminology.....	6
70	2.1Notational Conventions.....	6
71	2.2Namespaces.....	6
72	2.3Terminology.....	7
73	3Usage.....	8
74	3.1Processing Model.....	8
75	3.2SAML Version Differences.....	8
76	3.2.1Assertion Identifier.....	8
77	3.2.2Relationship of Subjects to Statements.....	8
78	3.2.3Assertion URI Reference Replaces AuthorityBinding.....	10
79	3.2.4Attesting Entity Identifier.....	10
80	3.3Attaching Security Tokens.....	10
81	3.4Identifying and Referencing Security Tokens.....	11
82	3.4.1SAML Assertion Referenced from Header or Element.....	13
83	3.4.2SAML Assertion Referenced from KeyInfo.....	14
84	3.4.3SAML Assertion Referenced from SignedInfo.....	16
85	3.4.4SAML Assertion Referenced from Encrypted Data Reference.....	17
86	3.4.5SAML Version Support and Backward Compatibility.....	17
87	3.5Subject Confirmation of SAML Assertions.....	17
88	3.5.1Holder-of-key Subject Confirmation Method.....	18
89	3.5.2Sender-vouches Subject Confirmation Method.....	22
90	3.5.3Bearer Confirmation Method.....	25
91	3.6Error Codes.....	26
92	4Threat Model and Countermeasures (non-normative).....	27
93	4.1Eavesdropping.....	27
94	4.2Replay.....	27
95	4.3Message Insertion.....	27
96	4.4Message Deletion.....	27
97	4.5Message Modification.....	27
98	4.6Man-in-the-Middle.....	28
99	5References	29
100	Appendix A. Acknowledgments.....	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

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

2.1 Notational Conventions

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](#).

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

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

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.

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

2.2 Namespaces

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:

Prefix	Namespace
s11	http://schemas.xmlsoap.org/soap/envelope/
s12	http://www.w3.org/2003/05/soap-envelope
ds	http://www.w3.org/2000/09/xmldsig#
xenc	http://www.w3.org/2001/04/xmenc#
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
wsse11	http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd
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
xsi	http://www.w3.org/2001/XMLSchema-instance

Table-1 Namespace Prefixes

140 **2.3 Terminology**

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

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.

3 Usage

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

3.1 Processing Model

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

When a receiver processes a `<wsse:Security>` header containing or referencing SAML assertions, it selects, based on its policy, the signatures and assertions that it will process. It is assumed that a receiver's signature selection policy MAY rely on semantic labeling¹ of `<wsse:SecurityTokenReference>` elements occurring in the `<ds:KeyInfo>` elements within the signatures. It is also assumed that the assertions selected for validation and processing will include those referenced from the `<ds:KeyInfo>` and `<ds:SignedInfo>` elements of the selected signatures.

As part of its validation and processing of the selected assertions, the receiver MUST² establish the relationship between the subject and claims of the SAML statements (of the referenced SAML assertions) and the entity providing the evidence to satisfy the confirmation method defined for the statements (i.e., the attesting entity). Two methods for establishing this correspondence, `holder-of-key` and `sender-vouches` are described below. Systems implementing this specification MUST implement the processing necessary to support both of these subject confirmation methods.

3.2 SAML Version Differences

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

3.2.1 Assertion Identifier

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

3.2.2 Relationship of Subjects to Statements

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

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

¹ The optional `Usage` attribute of the `<wsse:SecurityTokenReference>` element MAY be used to associate one or 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.

² 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.

³ 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 xmlns:saml="..." xmlns:ds="..."  
194   MajorVersion="1" MinorVersion="1" >  
195   ...  
196   <saml:SubjectStatement>  
197     <saml:Subject>  
198       <saml:NameIdentifier>  
199         ...  
200       </saml:NameIdentifier>  
201       <saml:SubjectConfirmation>  
202         <saml:ConfirmationMethod>  
203           urn:oasis:names:tc:SAML:1.0:cm:sender-vouches  
204         </saml:ConfirmationMethod>  
205         <saml:ConfirmationMethod>  
206           urn:oasis:names:tc:SAML:1.0:cm:holder-of-key  
207         </saml:ConfirmationMethod>  
208         <ds:KeyInfo>  
209           <ds:KeyValue>...</ds:KeyValue>  
210         </ds:KeyInfo>  
211       </saml:SubjectConfirmation>  
212     </saml:Subject>  
213     ... .  
214   </saml:SubjectStatement>  
215   <saml:SubjectStatement>  
216     <saml:Subject>  
217       <saml:NameIdentifier>  
218         ...  
219       </saml:NameIdentifier>  
220       <saml:SubjectConfirmation>  
221         <saml:ConfirmationMethod>  
222           urn:oasis:names:tc:SAML:1.0:cm:sender-vouches  
223         </saml:ConfirmationMethod>  
224       </saml:SubjectConfirmation>  
225     </saml:Subject>  
226     ... .  
227   </saml:SubjectStatement>  
228   ...  
229 </saml:Assertion>
```

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

```
243 <saml2:Assertion xmlns:saml2="..." xmlns:ds="..." xmlns:xsi="...">  
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 xmlns:saml2="..."
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 xmlns:S12="...">
303   <S12:Header>
304     <wsse:Security xmlns:wsse="...">
305       <saml:Assertion xmlns:saml="..."
306         AssertionID="a75adf55-01d7-40cc-929f-dbd8372ebdfc"
307         IssueInstant="2003-04-17T00:46:02Z"
308         Issuer="www.opensaml.org"
309         MajorVersion="1"
310         MinorVersion="1">
311         <saml:AuthenticationStatement>
312           <saml:Subject>
313             <saml:NameIdentifier
314               NameQualifier="www.example.com"
315               Format="urn:oasis:names:tc:SAML:1.1:nameid-
316 format:X509SubjectName">
317               uid=joe,ou=people,ou=saml-demo,o=baltimore.com
318             </saml:NameIdentifier>
319             <saml:SubjectConfirmation>
320               <saml:ConfirmationMethod>
321                 urn:oasis:names:tc:SAML:1.0:cm:bearer
322               </saml:ConfirmationMethod>
323             </saml:SubjectConfirmation>
324           </saml:Subject>
325         </saml:AuthenticationStatement>
326       </saml:Assertion>
327     </wsse:Security>
328   </S12:Header>
329   <S12:Body>
330     . . .
331   </S12:Body>
332 </S12:Envelope>

```

3.4 Identifying and Referencing Security Tokens

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

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

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

When a key identifier is used to reference a V1.1 SAML assertion that is not contained in the same message as the key identifier, a `<saml:AuthorityBinding>` element **MUST** be contained in the `<wsse:SecurityTokenReference>` element containing the key identifier. The contents of the `<saml:AuthorityBinding>` element **MUST** contain values sufficient for the intended recipients of

⁴ "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 the `<wsse:SecurityTokenReference>` to acquire the identified assertion from the intended
355 Authority. To this end, the value of the `AuthorityKind` attribute of the
356 `<saml:AuthorityBinding>` element MUST be "samlp:AssertionIdReference".

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

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

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

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

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

- 374 • An Embedded reference – a reference that encapsulates a security token.

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

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

- 379 • A SAML assertion may be referenced directly from a `<wsse:Security>` header element. In this
380 case, the assertion is being conveyed by reference in the message.
- 381 • A SAML assertion may be referenced from a `<ds:KeyInfo>` element of a `<ds:Signature>`
382 element in a `<wsse:Security>` header. In this case, the assertion contains a
383 `SubjectConfirmation` element that identifies the key used in the signature calculation.
- 384 • A SAML assertion reference may be referenced from a `<ds:Reference>` element within the
385 `<ds:SignedInfo>` element of a `<ds:Signature>` element in a `<wsse:Security>` header. In this
386 case, the doubly-referenced assertion is signed by the containing signature.
- 387 • A SAML assertion reference may occur as encrypted content within an `<xenc:EncryptedData>`
388 element referenced from a `<xenc:DataReference>` element within an `<xenc:ReferenceList>`
389 element. In this case, the assertion reference (which may contain an embedded assertion) is
390 encrypted.

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

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

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

402 Every key identifier, direct, or embedded reference to a SAML assertion SHOULD contain a
403 `wsse11:TokenType` attribute and the value of this attribute MUST be the value from Table 3 that

404 identifies the type and version of the referenced security token. When the referenced assertion is a SAML
 405 V2.0 Assertion the reference MUST contain a `wss11:TokenType` attribute (as described above).

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

406 Table-2 Key Identifier ValueType Attribute Values

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

407 Table-3 TokenType Attribute Values

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

411 3.4.1 SAML Assertion Referenced from Header or Element

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

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

```

419 <S12:Envelope xmlns:S12="...">
420   <S12:Header>
421     <wsse:Security xmlns:wsse="..." xmlns:wsu="..." xmlns:wss11="...">
422       <saml:Assertion xmlns:saml="..."
423         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
424         IssueInstant="2003-04-17T00:46:02Z"
425         Issuer="www.opensaml.org"
426         MajorVersion="1"
427         MinorVersion="1">
428       </saml:Assertion>
429       <wsse:SecurityTokenReference wsu:Id="STR1"
430         wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
431 profile-1.1#SAMLV1.1">
432         <wsse:KeyIdentifier wsu:Id="..."
433           ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
434 profile-1.0#SAMLAssertionID">
435           a75adf55-01d7-40cc-929f-dbd8372ebdfc
436         </wsse:KeyIdentifier>
437       </wsse:SecurityTokenReference>
438     </wsse:Security>
439   </S12:Header>
440   <S12:Body>
441     . . .
442   </S12:Body>
443 </S12:Envelope>
  
```

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

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

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

```
461 <wsse:SecurityTokenReference
462   xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..."
463   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 xmlns:saml="..."
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   <wsse:KeyIdentifier
471     wsu:Id="..."
472     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
473 1.0#SAMLAssertionID">
474     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
475   </wsse:KeyIdentifier>
476 </wsse:SecurityTokenReference>
```

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

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

```

502     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
503 1.0#SAMLAssertionID">
504     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
505     </wsse:KeyIdentifier>
506     </wsse:SecurityTokenReference>
507 </ds:KeyInfo>

```

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

```

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

```

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

```

528 <ds:KeyInfo xmlns:ds="...">
529   <wsse:SecurityTokenReference
530     xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..."
531     wsu:Id="STR1"
532     wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
533 profile-1.1#SAMLV1.1">
534     <saml:AuthorityBinding xmlns:saml="..."
535       Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
536       Location="http://www.opensaml.org/SAML-Authority"
537       AuthorityKind="samlp:AssertionIdReference"/>
538     <wsse:KeyIdentifier wsu:Id="..."
539       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
540 1.0#SAMLAssertionID">
541       _a75adf55-01d7-40cc-929f-dbd8372ebdfc
542     </wsse:KeyIdentifier>
543   </wsse:SecurityTokenReference>
544 </ds:KeyInfo>

```

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

```

547 <ds:KeyInfo xmlns:ds="...">
548   <wsse:SecurityTokenReference
549     xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..."
550     wsu:id="STR1"
551     wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
552 profile-1.1#SAMLV2.0">
553     <wsse:Reference
554       wsu:id="..."
555       URI="https://saml.example.edu/assertion-authority?ID=abcde">
556     </wsse:Reference>
557   </wsse:SecurityTokenReference>
558 </ds:KeyInfo>

```

559 `<ds:KeyInfo>` elements may also occur in `<xenc:EncryptedData>` and `<xenc:EncryptedKey>`
560 elements where they serve to identify the encryption key. `<ds:KeyInfo>` elements may also occur in

561 SAML SubjectConfirmation elements where they identify a key that MUST be demonstrated to
562 confirm the subject of the corresponding statement(s).
563 Conformant implementations of this profile are NOT required to process SAML assertion references
564 occurring within the <ds:KeyInfo> elements within <xenc:EncryptedData>,
565 <xenc:EncryptedKey>, or SAML SubjectConfirmation elements.

566 3.4.3 SAML Assertion Referenced from SignedInfo

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

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

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

```
583 <wsse:SecurityTokenReference
584   xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..." wsu:Id="STR1"
585   wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
586   profile-1.1#SAMLV1.1">
587   <saml:AuthorityBinding xmlns:saml="..."
588     Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
589     Location="http://www.opensaml.org/SAML-Authority"
590     AuthorityKind="samlp:AssertionIdReference"/>
591   <wsse:KeyIdentifier wsu:Id="..."
592     ValueType="http://docs.oasis-open.org/wss/oasis-2004XX-wss-saml-token-
593   profile-1.0#SAMLAssertionID">
594     a75adf55-01d7-40cc-929f-dbd8372ebdfc
595   </wsse:KeyIdentifier>
596 </wsse:SecurityTokenReference>
597
598 <ds:SignedInfo xmlns:ds="..." xmlns:wsse="...">
599   <ds:CanonicalizationMethod
600     Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
601   <ds:SignatureMethod
602     Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
603   <ds:Reference URI="#STR1">
604     <Transforms>
605       <ds:Transform
606         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
607   soap-message-security-1.0#STR-Transform">
608         <wsse:TransformationParameters>
609           <ds:CanonicalizationMethod
610             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
611           </wsse:TransformationParameters>
612         </ds:Transform>
613       </Transforms>
614       <ds:DigestMethod
615         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
616       <ds:DigestValue>...</ds:DigestValue>
617     </ds:Reference>
618   </ds:SignedInfo>
```


619 </ds:SignedInfo>

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

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

628 3.4.4 SAML Assertion Referenced from Encrypted Data Reference

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

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

```
641 <xenc:EncryptedData xmlns:xenc="..." xmlns:ds="..." Id="EncryptedSTR1">  
642   <ds:KeyInfo>  
643     . . .  
644   </ds:KeyInfo>  
645   <xenc:CipherData>  
646     <xenc:CipherValue>...</xenc:CipherValue>  
647   </xenc:CipherData>  
648 </xenc:EncryptedData>  
649 <xenc:ReferenceList xmlns:xenc="...">  
650   <xenc:DataReference URI="#EncryptedSTR1"/>  
651 </xenc:ReferenceList>
```

652 3.4.5 SAML Version Support and Backward Compatibility

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

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

662 3.5 Subject Confirmation of SAML Assertions

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

668 Any processor of SAML assertions MUST conform to the required validation and processing rules defined
 669 in the corresponding SAML specification including the validation of assertion signatures, the processing of
 670 <saml:Condition> elements within assertions, and the processing of
 671 <saml2:SubjectConfirmationData> attributes. [SAMLCoreV1] defines the validation and
 672 processing rules for V1.1 assertions, while [SAMLCoreV2] is authoritative for V2.0 assertions.
 673 The following table enumerates the mandatory subject confirmation methods and summarizes their
 674 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 SubjectConfirmation 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 MUST have an existing trust relationship with the attesting entity. The attesting entity MUST protect the assertion in combination with the message content against modification by another party. See also section 4.

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

680 3.5.1 Holder-of-key Subject Confirmation Method

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

684 3.5.1.1 Attesting Entity

685 An attesting entity demonstrates that it is authorized to act as the subject of a holder-of-key confirmed
 686 SAML statement by demonstrating knowledge of any key identified in a holder-of-key
 687 SubjectConfirmation element associated with the statement by the assertion containing the
 688 statement. Statements attested for by the holder-of-key method MUST be associated, within their
 689 containing assertion, with one or more holder-of-key SubjectConfirmation elements.

690 The SubjectConfirmation elements MUST include a <ds:KeyInfo> element that identifies a public
 691 or secret key⁵ that can be used to confirm the identity of the subject.

⁵[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 To satisfy the associated confirmation method processing to be performed by the message receiver, the
693 attesting entity MUST demonstrate knowledge of the confirmation key. The attesting entity MAY
694 accomplish this by using the confirmation key to sign content within the message and by including the
695 resulting <ds:Signature> element in the <wsse:Security> header. <ds:Signature> elements
696 produced for this purpose MUST conform to the canonicalization and token pre-pending rules defined in
697 the [WSS: SOAP Message Security](#) specification.

698 The attesting entity MAY protect against substitution of a different but equivalently confirmed⁶ assertion by
699 including, as described in section 3.4.3 "SAML Assertion Referenced from SignedInfo", the SAML
700 assertion (or an unambiguous reference to it) in the content signed to demonstrate knowledge of the
701 confirmation key.

702 SAML assertions that contain a holder-of-key SubjectConfirmation element SHOULD contain a
703 <ds:Signature> element that protects the integrity of the confirmation <ds:KeyInfo> established by
704 the assertion authority.

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

709 3.5.1.2 Receiver

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

714 If the receiver determines that the attesting entity has demonstrated knowledge of a subject confirmation
715 key, then the subjects and claims of the SAML statements confirmed by the key MAY be attributed to the
716 attesting entity and any content of the message ([including any SAML statements](#)) whose integrity is
717 protected by the key MAY be considered to have been provided by the attesting entity.

718 3.5.1.3 Example V1.1

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

```
722 <?xml version="1.0" encoding="UTF-8"?>
723 <S12:Envelope xmlns:S12="..." xmlns:wssu="...">
724   <S12:Header>
725
726     <wsse:Security xmlns:wssu="..." xmlns:wssell="..." xmlns:ds="...">
727       <saml:Assertion xmlns:saml="..."
728         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
729         IssueInstant="2005-05-27T16:53:33.173Z"
730         Issuer="www.opensaml.org"
731         MajorVersion="1"
732         MinorVersion="1">
733         <saml:Conditions
734           NotBefore="2005-05-27T16:53:33.173Z"
735           NotOnOrAfter="2005-05-27T16:58:33.17302Z"/>
736         <saml:AttributeStatement>
737           <saml:Subject>
738             <saml:NameIdentifier
739               NameQualifier="www.example.com"
740               Format="urn:oasis:names:tc:SAML:1.1:nameid-
741 format:X509SubjectName">
742               uid=joe,ou=people,ou=saml-demo,o=baltimore.com
```

⁶Two holder-of-key confirmed assertions are equivalently confirmed if they may be confirmed using the same confirmation key.

```

743         </saml:NameIdentifier>
744         <saml:SubjectConfirmation>
745             <saml:ConfirmationMethod>
746                 urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
747             </saml:ConfirmationMethod>
748             <ds:KeyInfo>
749                 <ds:KeyValue>...</ds:KeyValue>
750             </ds:KeyInfo>
751         </saml:SubjectConfirmation>
752     </saml:Subject>
753     <saml:Attribute
754         AttributeName="MemberLevel"
755         AttributeNamespace="http://www.oasis-
756 open.org/Catalyst2002/attributes">
757         <saml:AttributeValue>gold</saml:AttributeValue>
758     </saml:Attribute>
759     <saml:Attribute
760         AttributeName="E-mail"
761         AttributeNamespace="http://www.oasis-
762 open.org/Catalyst2002/attributes">
763         <saml:AttributeValue>joe@yahoo.com</saml:AttributeValue>
764     </saml:Attribute>
765 </saml:AttributeStatement>
766 <ds:Signature>...</ds:Signature>
767 </saml:Assertion>
768
769 <ds:Signature>
770 <ds:SignedInfo>
771 <ds:CanonicalizationMethod
772     Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
773 <ds:SignatureMethod
774     Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
775 <ds:Reference
776     URI="#MsgBody">
777     <ds:DigestMethod
778         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
779     <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
780 </ds:Reference>
781 </ds:SignedInfo>
782 <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
783 <ds:KeyInfo>
784     <wsse:SecurityTokenReference wsu:Id="STR1"
785         wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
786 token-profile-1.1#SAMLV1.1">
787         <wsse:KeyIdentifier wsu:Id="..."
788             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
789 profile-1.0#SAMLAssertionID">
790             a75adf55-01d7-40cc-929f-dbd8372ebdfc
791         </wsse:KeyIdentifier>
792     </wsse:SecurityTokenReference>
793 </ds:KeyInfo>
794 </ds:Signature>
795 </wsse:Security>
796 </S12:Header>
797
798 <S12:Body wsu:Id="MsgBody">
799     <ReportRequest>
800         <TickerSymbol>SUNW</TickerSymbol>
801     </ReportRequest>
802 </S12:Body>
803 </S12:Envelope>

```

804 3.5.1.4 Example V2.0

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

807 <wsse:Security> header:

```
808 <?xml version="1.0" encoding="UTF-8"?>
809 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
810   <S12:Header>
811     <wsse:Security xmlns:wsse="..." xmlns:wssell="..." xmlns:ds="...">
812       <saml2:Assertion xmlns:saml2="..." xmlns:xsi="..."
813         ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
814         <saml2:Subject>
815           <saml2:NameID>
816             ...
817           </saml2:NameID>
818           <saml2:SubjectConfirmation
819             Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
820             <saml2:SubjectConfirmationData
821               xsi:type="saml2:KeyInfoConfirmationDataType">
822               <ds:KeyInfo>
823                 <ds:KeyValue>...</ds:KeyValue>
824               </ds:KeyInfo>
825             </saml2:SubjectConfirmationData>
826           </saml2:SubjectConfirmation>
827         </saml2:Subject>
828         <saml2:Statement>
829           ...
830         </saml2:Statement>
831         <ds:Signature>...</ds:Signature>
832       </saml2:Assertion>
833     <ds:Signature>
834       <ds:SignedInfo>
835         <ds:CanonicalizationMethod
836           Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
837         <ds:SignatureMethod
838           Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
839         <ds:Reference
840           URI="#MsgBody">
841           <ds:DigestMethod
842             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
843           <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
844         </ds:Reference>
845       </ds:SignedInfo>
846       <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
847       <ds:KeyInfo>
848         <wsse:SecurityTokenReference wsu:Id="STR1"
849           wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
850 token-profile-1.1#SAMLV2.0">
851         <wsse:KeyIdentifier wsu:Id="..."
852           ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
853 profile-1.1#SAMLID">
854           _a75adf55-01d7-40cc-929f-dbd8372ebdfc
855         </wsse:KeyIdentifier>
856       </wsse:SecurityTokenReference>
857     </ds:KeyInfo>
858   </ds:Signature>
859 </wsse:Security>
860 </S12:Header>
861 <S12:Body wsu:Id="MsgBody">
862   <ReportRequest>
863     <TickerSymbol>SUNW</TickerSymbol>
864   </ReportRequest>
```

```
868     </S12:Body>
869 </S12:Envelope>
```

870 3.5.2 Sender-vouches Subject Confirmation Method

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

874 3.5.2.1 Attesting Entity

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

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

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

889 3.5.2.2 Receiver

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

895 3.5.2.3 Example V1.1

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

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

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

```
907 <?xml version="1.0" encoding="UTF-8"?>
908 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
909
910   <S12:Header>
911     <wsse:Security xmlns:wsse="..." xmlns:wsse11="..." xmlns:ds="...">
912
913       <saml:Assertion xmlns:saml="..."
914         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
915         IssueInstant="2005-05-27T16:53:33.173Z"
916         Issuer="www.opensaml.org">
```

```

917     MajorVersion="1"
918     MinorVersion="1">
919     <saml:Conditions
920         NotBefore="2005-05-27T16:53:33.173Z"
921         NotOnOrAfter="2005-05-27T16:58:33.173Z"/>
922     <saml:AttributeStatement>
923         <saml:Subject>
924             <saml:NameIdentifier
925                 NameQualifier="www.example.com"
926                 Format="urn:oasis:names:tc:SAML:1.1:nameid-
927 format:X509SubjectName">
928                 uid=proxy,ou=system,ou=saml-demo,o=baltimore.com
929             </saml:NameIdentifier>
930             <saml:SubjectConfirmation>
931                 <saml:ConfirmationMethod>
932                     urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
933                 </saml:ConfirmationMethod>
934                 <ds:KeyInfo>
935                     <ds:KeyValue>...</ds:KeyValue>
936                 </ds:KeyInfo>
937             </saml:SubjectConfirmation>
938         </saml:Subject>
939         <saml:Attribute>
940             .
941             .
942             .
943         </saml:AttributeStatement>
944     </saml:Assertion>
945
946     <wsse:SecurityTokenReference wsu:Id="STR1">
947         wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
948 profile-1.1#SAMLV1.1">
949         <saml:AuthorityBinding xmlns:saml="..."
950             Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
951             Location="http://www.opensaml.org/SAML-Authority"
952             AuthorityKind="samlp:AssertionIdReference"/>
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     wsse11: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 xmlns:S12="..." xmlns:wsu="...">
1013     <S12:Header>
1014
1015         <wsse:Security xmlns:wsse="..." xmlns:wsse11="..." xmlns:ds="...">
1016             <saml2:Assertion xmlns:saml2="..." xmlns:xsi="..."
1017
1018                 ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
1019                 <saml2:Subject>
1020                 <saml2:NameID>
1021                     ...
1022                 </saml2:NameID>
1023                 <saml2:SubjectConfirmation
1024                     Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
1025                 <saml2:SubjectConfirmationData
1026                     xsi:type="saml2:KeyInfoConfirmationDataType">
1027                     <ds:KeyInfo>
1028                         <ds:KeyValue>...</ds:KeyValue>
1029                     </ds:KeyInfo>
1030                 </saml2:SubjectConfirmationData>
1031                 </saml2:SubjectConfirmation>
1032                 </saml2:Subject>
1033                 <saml2:Statement>
1034                     ...
1035                 </saml2:Statement>
1036                 <ds:Signature>...</ds:Signature>
1037             </saml2:Assertion>
1038
1039             <wsse:SecurityTokenReference wsu:Id="STR1"
1040             wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1041 profile-1.1#SAMLV2.0">
1042             <wsse:Reference wsu:Id="..."
1043                 URI="https://www.opensaml.org?_a75adf55-01d7-40cc-929f-
1044 dbd8372ebdbe">
1045             </wsse:Reference>

```



```

1046     </wsse:SecurityTokenReference>
1047
1048     <ds:Signature>
1049       <ds:SignedInfo>
1050         <ds:CanonicalizationMethod
1051           Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1052         <ds:SignatureMethod
1053           Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
1054         <ds:Reference URI="#STR1">
1055           <Transforms>
1056             <ds:Transform
1057               Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
1058 wss-soap-message-security-1.0#STR-Transform">
1059               <wsse:TransformationParameters>
1060                 <ds:CanonicalizationMethod
1061                   Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1062                 </wsse:TransformationParameters>
1063               </ds:Transform>
1064             </Transforms>
1065           </ds:SignatureMethod>
1066           <ds:DigestMethod
1067             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1068           <ds:DigestValue>...</ds:DigestValue>
1069         </ds:Reference>
1070         <ds:Reference URI="#MsgBody">
1071           <ds:DigestMethod
1072             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1073           <ds:DigestValue>...</ds:DigestValue>
1074         </ds:Reference>
1075       </ds:SignedInfo>
1076       <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
1077       <ds:KeyInfo>
1078         <wsse:SecurityTokenReference wsu:Id="STR2">
1079           wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
1080 token-profile-1.1#SAMLV2.0">
1081           <wsse:KeyIdentifier wsu:Id="..."
1082             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1083 profile-1.1#SAMLID">
1084             a75adf55-01d7-40cc-929f-dbd8372ebdfc
1085           </wsse:KeyIdentifier>
1086         </wsse:SecurityTokenReference>
1087       </ds:KeyInfo>
1088     </ds:Signature>
1089   </wsse:Security>
1090 </S12:Header>
1091
1092 <S12:Body wsu:Id="MsgBody">
1093   <ReportRequest>
1094     <TickerSymbol>SUNW</TickerSymbol>
1095   </ReportRequest>
1096 </S12:Body>
1097 </S12:Envelope>

```

1098 3.5.3 Bearer Confirmation Method

1099 This profile does NOT require message receivers to establish the relationship between a received
1100 message and the statements of any bearer confirmed (i.e., confirmation method
1101 urn:oasis:names:tc:SAML:1.0:cm:bearer) assertions conveyed or referenced from the message.
1102 Conformant implementations of this profile MUST be able to process references and convey bearer
1103 assertions within <wsse:Security> headers. Any additional processing requirements that pertain
1104 specifically to bearer confirmed assertions are outside the scope of this profile.

1105 **3.6 Error Codes**

1106 When a system that implements the SAML token profile of [WSS: SOAP Message Security](#) does not
 1107 perform its normal processing because of an error detected during the processing of a security header, it
 1108 MAY choose to report the cause of the error using the SOAP fault mechanism. The SAML token profile of
 1109 [WSS: SOAP Message Security](#) does not require that SOAP faults be returned for such errors, and
 1110 systems that choose to return faults SHOULD take care not to introduce any security vulnerabilities as a
 1111 result of the information returned in error responses.

1112 Systems that choose to return faults SHOULD respond with the error codes and fault strings defined in the
 1113 [WSS: SOAP Message Security](#) specification. The RECOMMENDED correspondence between the
 1114 common assertion processing failures and the error codes defined in [WSS: SOAP Message Security](#) are
 1115 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

1116 The preceding table defines fault codes in a form suitable for use with SOAP 1.1. The [WSS: SOAP](#)
 1117 [Message Security](#) specification describes how to map SOAP 1.1 fault constructs to the SOAP 1.2 fault
 1118 constructs.

1119 4 Threat Model and Countermeasures (non- 1120 normative)

1121 This document defines the mechanisms and procedures for securely attaching SAML assertions to SOAP
1122 messages. SOAP messages are used in multiple contexts, specifically including cases where the
1123 message is transported without an active session, the message is persisted, or the message is routed
1124 through a number of intermediaries. Such a general context of use suggests that users of this profile must
1125 be concerned with a variety of threats.

1126 In general, the use of SAML assertions with [WSS: SOAP Message Security](#) introduces no new threats
1127 beyond those identified for SAML or by the [WSS: SOAP Message Security](#) specification. The following
1128 sections provide an overview of the characteristics of the threat model, and the countermeasures that
1129 SHOULD be adopted for each perceived threat.

1130 4.1 Eavesdropping

1131 Eavesdropping is a threat to the SAML token profile of [WSS: SOAP Message Security](#) in the same
1132 manner as it is a threat to any network protocol. The routing of SOAP messages through intermediaries
1133 increases the potential incidences of eavesdropping. Additional opportunities for eavesdropping exist
1134 when SOAP messages are persisted.

1135 To provide maximum protection from eavesdropping, assertions, assertion references, and sensitive
1136 message content SHOULD be encrypted such that only the intended audiences can view their content.
1137 This approach removes threats of eavesdropping in transit, but MAY not remove risks associated with
1138 storage or poor handling by the receiver.

1139 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1140 references from eavesdropping while in transport, but message content MUST be encrypted above the
1141 transport if it is to be protected from eavesdropping by intermediaries.

1142 4.2 Replay

1143 Reliance on authority-protected (e.g., signed) assertions with a holder-of-key subject confirmation
1144 mechanism precludes all but a holder of the key from binding the assertions to a SOAP message.
1145 Although this mechanism effectively restricts data origin to a holder of the confirmation key, it does not, by
1146 itself, provide the means to detect the capture and resubmission of the message by other parties.

1147 Assertions that contain a sender-vouches confirmation mechanism introduce another dimension to replay
1148 vulnerability if the assertions impose no restriction on the entities that may use or reuse the assertions.

1149 Replay attacks can be detected by receivers if message senders include additional message identifying
1150 information (e.g., timestamps, nonces, and or recipient identifiers) within origin-protected message
1151 content and receivers check this information against previously received values.

1152 4.3 Message Insertion

1153 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message insertion attacks.

1154 4.4 Message Deletion

1155 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message deletion attacks.

1156 4.5 Message Modification

1157 Messages constructed according to this specification are protected from message modification if receivers
1158 can detect unauthorized modification of relevant message content. Therefore, it is strongly
1159 RECOMMENDED that all relevant and immutable message content be signed by an attesting entity.
1160 Receivers SHOULD only consider the correspondence between the subject of the SAML assertions and

1161 the SOAP message content to have been established for those portions of the message that are protected
1162 by the attesting entity against modification by another entity.

1163 To ensure that message receivers can have confidence that received assertions have not been forged or
1164 altered since their issuance, SAML assertions appearing in or referenced from <wsse:Security>
1165 header elements MUST be protected against unauthorized modification (e.g., signed) by their issuing
1166 authority or the attesting entity (as the case warrants). It is strongly RECOMMENDED that an attesting
1167 entity sign any <saml:Assertion> elements that it is attesting for and that are not signed by their
1168 issuing authority.

1169 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1170 assertion references from modification while in transport, but signatures are required to extend such
1171 protection through intermediaries.

1172 To ensure that message receivers can have confidence that an assertion with an equivalent confirmation
1173 key has not been substituted for the assertion used by the attesting entity, the attesting entity MAY include
1174 the assertion (or an unambiguous reference to it) in the attested for (i.e., signed) message content.

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. Acknowledgments

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
Sundeep	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
--------	-----------	--------