



Web Services Security: SAML Token Profile 1.1

OASIS Standard, 1 February 2006

Document Identifier:

[wss-v1.1-spec-os-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. It 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

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

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

46 Copyright (C) OASIS Open 2002-2006. All Rights Reserved.

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

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

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

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

Table of Contents

64	1 Introduction.....	4
65	1.1 Goals.....	4
66	1.1.1 Non-Goals.....	4
67	2 Notations and Terminology.....	5
68	2.1 Notational Conventions.....	5
69	2.2 Namespaces.....	5
70	2.3 Terminology.....	6
71	3 Usage.....	7
72	3.1 Processing Model.....	7
73	3.2 SAML Version Differences.....	7
74	3.2.1 Assertion Identifier.....	7
75	3.2.2 Relationship of Subjects to Statements.....	7
76	3.2.3 Assertion URI Reference Replaces AuthorityBinding.....	9
77	3.2.4 Attesting Entity Identifier.....	9
78	3.3 Attaching Security Tokens.....	9
79	3.4 Identifying and Referencing Security Tokens.....	10
80	3.4.1 SAML Assertion Referenced from Header or Element.....	12
81	3.4.2 SAML Assertion Referenced from KeyInfo.....	13
82	3.4.3 SAML Assertion Referenced from SignedInfo.....	15
83	3.4.4 SAML Assertion Referenced from Encrypted Data Reference.....	16
84	3.4.5 SAML Version Support and Backward Compatibility.....	16
85	3.5 Subject Confirmation of SAML Assertions.....	16
86	3.5.1 Holder-of-key Subject Confirmation Method.....	17
87	3.5.2 Sender-vouches Subject Confirmation Method.....	21
88	3.5.3 Bearer Confirmation Method.....	24
89	3.6 Error Codes.....	25
90	4 Threat Model and Countermeasures (non-normative).....	26
91	4.1 Eavesdropping.....	26
92	4.2 Replay.....	26
93	4.3 Message Insertion.....	26
94	4.4 Message Deletion.....	26
95	4.5 Message Modification.....	26
96	4.6 Man-in-the-Middle.....	27
97	5 References	28
98	Appendix B. Acknowledgments.....	29
99		

100 **1 Introduction**

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

105 **1.1 Goals**

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

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

112 **1.1.1 Non-Goals**

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

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

138 **2.3 Terminology**

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

141 Attesting Entity – the entity that provides the confirmation evidence that will be used to establish the
142 correspondence between the subjects and claims of SAML statements (in SAML assertions) and SOAP
143 message content.

144 Confirmation Method Identifier – the value within a SAML SubjectConfirmation element that identifies the
145 subject confirmation process to be used with the corresponding statements.

146 Subject Confirmation – the process of establishing the correspondence between the subject and claims of
147 SAML statements (in SAML assertions) and SOAP message content by verifying the confirmation
148 evidence provided by an attesting entity.

149 SAML Assertion Authority - A *system entity* that issues *assertions*.

150 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.

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

```
191 <saml:Assertion xmlns:saml="..." xmlns:ds="..."  
192   MajorVersion="1" MinorVersion="1" >  
193   ...  
194   <saml:SubjectStatement>  
195     <saml:Subject>  
196       <saml:NameIdentifier>  
197         ...  
198       </saml:NameIdentifier>  
199       <saml:SubjectConfirmation>  
200         <saml:ConfirmationMethod>  
201           urn:oasis:names:tc:SAML:1.0:cm:sender-vouches  
202         </saml:ConfirmationMethod>  
203         <saml:ConfirmationMethod>  
204           urn:oasis:names:tc:SAML:1.0:cm:holder-of-key  
205         </saml:ConfirmationMethod>  
206         <ds:KeyInfo>  
207           <ds:KeyValue>...</ds:KeyValue>  
208         </ds:KeyInfo>  
209       </saml:SubjectConfirmation>  
210     </saml:Subject>  
211     ... .  
212   </saml:SubjectStatement>  
213   <saml:SubjectStatement>  
214     <saml:Subject>  
215       <saml:NameIdentifier>  
216         ...  
217       </saml:NameIdentifier>  
218       <saml:SubjectConfirmation>  
219         <saml:ConfirmationMethod>  
220           urn:oasis:names:tc:SAML:1.0:cm:sender-vouches  
221         </saml:ConfirmationMethod>  
222       </saml:SubjectConfirmation>  
223     </saml:Subject>  
224     ... .  
225   </saml:SubjectStatement>  
226   ...  
227 </saml:Assertion>
```

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

```
241 <saml2:Assertion xmlns:saml2="..." xmlns:ds="..." xmlns:xsi="...">  
242   <saml2:Subject>  
243     <saml2:NameID>  
244       ...  
245     </saml2:NameID>  
246     <saml2:SubjectConfirmation
```



```

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

```

273 3.2.3 Assertion URI Reference Replaces AuthorityBinding

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

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

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

280 <https://saml.example.edu/assertion-authority>

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

282 <https://saml.example.edu/assertion-authority?ID=abcde>

283 3.2.4 Attesting Entity Identifier

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

```

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

```

295 3.3 Attaching Security Tokens

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

```

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

```

333 3.4 Identifying and Referencing Security Tokens

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

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

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

348 When a key identifier is used to reference a V1.1 SAML assertion that is not contained in the same
349 message as the key identifier, a `<saml:AuthorityBinding>` element **MUST** be contained in the
350 `<wsse:SecurityTokenReference>` element containing the key identifier. The contents of the
351 `<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).

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

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

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

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

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

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

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

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

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

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

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

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

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

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

402 identifies the type and version of the referenced security token. When the referenced assertion is a SAML
 403 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

404 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

405 Table-3 TokenType Attribute Values

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

409 3.4.1 SAML Assertion Referenced from Header or Element

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

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

```

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

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

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

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

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

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

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

487 3.4.2 SAML Assertion Referenced from KeyInfo

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

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

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

```

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

```

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

```

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

```

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

```

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

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

564 3.4.3 SAML Assertion Referenced from SignedInfo

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

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

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

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

```
614 <ds:DigestValue>...</ds:DigestValue>
615 </ds:Reference>
616 </ds:SignedInfo>
```

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

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

626 3.4.4 SAML Assertion Referenced from Encrypted Data Reference

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

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

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

650 3.4.5 SAML Version Support and Backward Compatibility

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

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

660 3.5 Subject Confirmation of SAML Assertions

661 The SAML profile of [WSS: SOAP Message Security](#) requires that systems support the holder-of-key and
662 sender-vouches methods of subject confirmation. It is strongly RECOMMENDED that an XML signature
663 be used to establish the relationship between the message and the statements of the attached assertions.

664 This is especially RECOMMENDED whenever the SOAP message exchange is conducted over an
 665 unprotected transport.

666 Any processor of SAML assertions MUST conform to the required validation and processing rules defined
 667 in the corresponding SAML specification including the validation of assertion signatures, the processing of
 668 <saml:Condition> elements within assertions, and the processing of
 669 <saml2:SubjectConfirmationData> attributes. [SAMLCoreV1] defines the validation and
 670 processing rules for V1.1 assertions, while [SAMLCoreV2] is authoritative for V2.0 assertions.

671 The following table enumerates the mandatory subject confirmation methods and summarizes their
 672 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.

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

678 3.5.1 Holder-of-key Subject Confirmation Method

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

682 3.5.1.1 Attesting Entity

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

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

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

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

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

703 3.5.1.2 Receiver

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

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

712 3.5.1.3 Example V1.1

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

```
716 <?xml version="1.0" encoding="UTF-8"?>  
717 <S12:Envelope xmlns:S12="..." xmlns:wssu="...">  
718   <S12:Header>  
719     <wsse:Security xmlns:wsse="..." xmlns:wssell="..." xmlns:ds="...">  
720       <saml:Assertion xmlns:saml="...">  
721         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"  
722         IssueInstant="2005-05-27T16:53:33.173Z"  
723         Issuer="www.opensaml.org"  
724         MajorVersion="1"  
725         MinorVersion="1">  
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
```

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)`.

```

733         NameQualifier="www.example.com"
734         Format="urn:oasis:names:tc:SAML:1.1:nameid-
735 format:X509SubjectName">
736         uid=joe,ou=people,ou=saml-demo,o=baltimore.com
737     </saml:NameIdentifier>
738     <saml:SubjectConfirmation>
739         <saml:ConfirmationMethod>
740             urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
741         </saml:ConfirmationMethod>
742         <ds:KeyInfo>
743             <ds:KeyValue>...</ds:KeyValue>
744         </ds:KeyInfo>
745     </saml:SubjectConfirmation>
746 </saml:Subject>
747 <saml:Attribute
748     AttributeName="MemberLevel"
749     AttributeNamespace="http://www.oasis-
750 open.org/Catalyst2002/attributes">
751     <saml:AttributeValue>gold</saml:AttributeValue>
752 </saml:Attribute>
753 <saml:Attribute
754     AttributeName="E-mail"
755     AttributeNamespace="http://www.oasis-
756 open.org/Catalyst2002/attributes">
757     <saml:AttributeValue>joe@yahoo.com</saml:AttributeValue>
758 </saml:Attribute>
759 </saml:AttributeStatement>
760 <ds:Signature>...</ds:Signature>
761 </saml:Assertion>
762
763 <ds:Signature>
764     <ds:SignedInfo>
765         <ds:CanonicalizationMethod
766             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
767         <ds:SignatureMethod
768             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
769         <ds:Reference
770             URI="#MsgBody">
771             <ds:DigestMethod
772                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
773             <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
774         </ds:Reference>
775     </ds:SignedInfo>
776     <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
777     <ds:KeyInfo>
778         <wsse:SecurityTokenReference wsu:Id="STR1"
779             wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
780 token-profile-1.1#SAMLV1.1">
781             <wsse:KeyIdentifier wsu:Id="..."
782                 ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
783 profile-1.0#SAMLAssertionID">
784                 _a75adf55-01d7-40cc-929f-dbd8372ebdfc
785             </wsse:KeyIdentifier>
786         </wsse:SecurityTokenReference>
787     </ds:KeyInfo>
788 </ds:Signature>
789 </wsse:Security>
790 </S12:Header>
791
792 <S12:Body wsu:Id="MsgBody">
793     <ReportRequest>
794         <TickerSymbol>SUNW</TickerSymbol>
795     </ReportRequest>
796 </S12:Body>
797 </S12:Envelope>

```

798 3.5.1.4 Example V2.0

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

801 <wsse:Security> header:

```
802 <?xml version="1.0" encoding="UTF-8"?>
803 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
804   <S12:Header>
805
806     <wsse:Security xmlns:wsse="..." xmlns:wssell="..." xmlns:ds="...">
807       <saml2:Assertion xmlns:saml2="..." xmlns:xsi="..."
808         ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
809         <saml2:Subject>
810           <saml2:NameID>
811             ...
812           </saml2:NameID>
813           <saml2:SubjectConfirmation
814             Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
815             <saml2:SubjectConfirmationData
816               xsi:type="saml2:KeyInfoConfirmationDataType">
817               <ds:KeyInfo>
818                 <ds:KeyValue>...</ds:KeyValue>
819               </ds:KeyInfo>
820             </saml2:SubjectConfirmationData>
821             </saml2:SubjectConfirmation>
822           </saml2:Subject>
823           <saml2:Statement>
824             ...
825           </saml2:Statement>
826           <ds:Signature>...</ds:Signature>
827         </saml2:Assertion>
828
829         <ds:Signature>
830           <ds:SignedInfo>
831             <ds:CanonicalizationMethod
832               Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
833             <ds:SignatureMethod
834               Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
835             <ds:Reference
836               URI="#MsgBody">
837               <ds:DigestMethod
838                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
839               <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
840             </ds:Reference>
841             </ds:SignedInfo>
842             <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
843             <ds:KeyInfo>
844               <wsse:SecurityTokenReference wsu:Id="STR1"
845                 wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
846 token-profile-1.1#SAMLV2.0">
847                 <wsse:KeyIdentifier wsu:Id="..."
848                   ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
849 profile-1.1#SAMLID">
850                   _a75adf55-01d7-40cc-929f-dbd8372ebdfc
851                 </wsse:KeyIdentifier>
852               </wsse:SecurityTokenReference>
853             </ds:KeyInfo>
854           </ds:Signature>
855         </wsse:Security>
856       </S12:Header>
857
858       <S12:Body wsu:Id="MsgBody">
859         <ReportRequest>
860           <TickerSymbol>SUNW</TickerSymbol>
861         </ReportRequest>
```

```
862     </S12:Body>
863 </S12:Envelope>
```

864 3.5.2 Sender-vouches Subject Confirmation Method

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

868 3.5.2.1 Attesting Entity

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

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

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

883 3.5.2.2 Receiver

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

889 3.5.2.3 Example V1.1

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

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

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

```
901 <?xml version="1.0" encoding="UTF-8"?>
902 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
903
904   <S12:Header>
905     <wsse:Security xmlns:wsse="..." xmlns:wsse11="..." xmlns:ds="...">
906
907       <saml:Assertion xmlns:saml="..."
908         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
909         IssueInstant="2005-05-27T16:53:33.173Z"
910         Issuer="www.opensaml.org">
```

```

911     MajorVersion="1"
912     MinorVersion="1">
913     <saml:Conditions
914         NotBefore="2005-05-27T16:53:33.173Z"
915         NotOnOrAfter="2005-05-27T16:58:33.173Z"/>
916     <saml:AttributeStatement>
917         <saml:Subject>
918             <saml:NameIdentifier
919                 NameQualifier="www.example.com"
920                 Format="urn:oasis:names:tc:SAML:1.1:nameid-
921 format:X509SubjectName">
922                 uid=proxy,ou=system,ou=saml-demo,o=baltimore.com
923             </saml:NameIdentifier>
924             <saml:SubjectConfirmation>
925                 <saml:ConfirmationMethod>
926                     urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
927                 </saml:ConfirmationMethod>
928                 <ds:KeyInfo>
929                     <ds:KeyValue>...</ds:KeyValue>
930                 </ds:KeyInfo>
931             </saml:SubjectConfirmation>
932         </saml:Subject>
933         <saml:Attribute>
934             .
935             .
936             .
937         </saml:Attribute>
938     </saml:AttributeStatement>
939 </saml:Assertion>
940     <wsse:SecurityTokenReference wsu:Id="STR1">
941         wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
942 profile-1.1#SAMLV1.1">
943         <saml:AuthorityBinding xmlns:saml="..."
944             Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
945             Location="http://www.opensaml.org/SAML-Authority"
946             AuthorityKind="samlp:AssertionIdReference"/>
947         <wsse:KeyIdentifier wsu:Id="..."
948             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
949 profile-1.0#SAMLAssertionID">
950             _a75adf55-01d7-40cc-929f-dbd8372ebdbe
951         </wsse:KeyIdentifier>
952     </wsse:SecurityTokenReference>
953     <ds:Signature>
954         <ds:SignedInfo>
955             <ds:CanonicalizationMethod
956                 Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
957             <ds:SignatureMethod
958                 Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
959             <ds:Reference URI="#STR1">
960                 <Transforms>
961                     <ds:Transform
962                         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-
963 200401-wss-soap-message-security-1.0#STR-Transform">
964                         <wsse:TransformationParameters>
965                             <ds:CanonicalizationMethod
966                                 Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
967                             </wsse:TransformationParameters>
968                         </ds:Transform>
969                     </Transforms>
970                 </ds:Reference>
971             <ds:DigestMethod
972                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
973             <ds:DigestValue>...</ds:DigestValue>
974         </ds:Signature>
975     <ds:Reference URI="#MsgBody">
976         <ds:DigestMethod

```

```

977         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
978         <ds:DigestValue>...</ds:DigestValue>
979     </ds:Reference>
980 </ds:SignedInfo>
981 <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
982 <ds:KeyInfo>
983     <wsse:SecurityTokenReference wsu:Id="STR2"
984         wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
985 token-profile-1.1#SAMLV1.1">
986         <wsse:KeyIdentifier wsu:Id="..."
987             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
988 profile-1.0#SAMLAssertionID">
989             _a75adf55-01d7-40cc-929f-dbd8372ebdfc
990         </wsse:KeyIdentifier>
991     </wsse:SecurityTokenReference>
992 </ds:KeyInfo>
993 </ds:Signature>
994 </wsse:Security>
995 </S12:Header>
996
997 <S12:Body wsu:Id="MsgBody">
998     <ReportRequest>
999         <TickerSymbol>SUNW</TickerSymbol>
1000     </ReportRequest>
1001 </S12:Body>
1002 </S12:Envelope>

```

1003 3.5.2.4 Example V2.0

1004 The following example illustrates the mapping of the preceding example to SAML V2.0 assertions.

```

1005 <?xml version="1.0" encoding="UTF-8"?>
1006 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
1007     <S12:Header>
1008
1009         <wsse:Security xmlns:wsse="..." xmlns:wsse11="..." xmlns:ds="...">
1010             <saml2:Assertion xmlns:saml2="..." xmlns:xsi="..."
1011
1012                 ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
1013                 <saml2:Subject>
1014                     <saml2:NameID>
1015                         ...
1016                     </saml2:NameID>
1017                 <saml2:SubjectConfirmation
1018                     Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
1019                     <saml2:SubjectConfirmationData
1020                         xsi:type="saml2:KeyInfoConfirmationDataType">
1021                         <ds:KeyInfo>
1022                             <ds:KeyValue>...</ds:KeyValue>
1023                         </ds:KeyInfo>
1024                     </saml2:SubjectConfirmationData>
1025                     </saml2:SubjectConfirmation>
1026                 </saml2:Subject>
1027                 <saml2:Statement>
1028                     ...
1029                 </saml2:Statement>
1030                 <ds:Signature>...</ds:Signature>
1031             </saml2:Assertion>
1032
1033             <wsse:SecurityTokenReference wsu:Id="STR1"
1034                 wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1035 profile-1.1#SAMLV2.0">
1036                 <wsse:Reference wsu:Id="..."
1037                     URI="https://www.opensaml.org?_a75adf55-01d7-40cc-929f-
1038 dbd8372ebdbe">
1039                 </wsse:Reference>

```

```

1040     </wsse:SecurityTokenReference>
1041
1042     <ds:Signature>
1043       <ds:SignedInfo>
1044         <ds:CanonicalizationMethod
1045           Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1046         <ds:SignatureMethod
1047           Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
1048         <ds:Reference URI="#STR1">
1049           <Transforms>
1050             <ds:Transform
1051
1052               Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
1053 wss-soap-message-security-1.0#STR-Transform">
1054               <wsse:TransformationParameters>
1055                 <ds:CanonicalizationMethod
1056                   Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1057                 </wsse:TransformationParameters>
1058               </ds:Transform>
1059             </Transforms>
1060             <ds:DigestMethod
1061               Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1062             <ds:DigestValue>...</ds:DigestValue>
1063           </ds:Reference>
1064         <ds:Reference URI="#MsgBody">
1065           <ds:DigestMethod
1066             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1067           <ds:DigestValue>...</ds:DigestValue>
1068         </ds:Reference>
1069       </ds:SignedInfo>
1070       <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
1071       <ds:KeyInfo>
1072         <wsse:SecurityTokenReference wsu:Id="STR2">
1073           wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
1074 token-profile-1.1#SAMLV2.0">
1075           <wsse:KeyIdentifier wsu:Id="..."
1076             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1077 profile-1.1#SAMLID">
1078             a75adf55-01d7-40cc-929f-dbd8372ebdfc
1079           </wsse:KeyIdentifier>
1080         </wsse:SecurityTokenReference>
1081       </ds:KeyInfo>
1082     </ds:Signature>
1083   </wsse:Security>
1084 </S12:Header>
1085
1086 <S12:Body wsu:Id="MsgBody">
1087   <ReportRequest>
1088     <TickerSymbol>SUNW</TickerSymbol>
1089   </ReportRequest>
1090 </S12:Body>
1091 </S12:Envelope>

```

1092 3.5.3 Bearer Confirmation Method

1093 This profile does NOT require message receivers to establish the relationship between a received
1094 message and the statements of any bearer confirmed (i.e., confirmation method
1095 urn:oasis:names:tc:SAML:1.0:cm:bearer) assertions conveyed or referenced from the message.
1096 Conformant implementations of this profile MUST be able to process references and convey bearer
1097 assertions within <wsse:Security> headers. Any additional processing requirements that pertain
1098 specifically to bearer confirmed assertions are outside the scope of this profile.

1099 **3.6 Error Codes**

1100 When a system that implements the SAML token profile of [WSS: SOAP Message Security](#) does not
 1101 perform its normal processing because of an error detected during the processing of a security header, it
 1102 MAY choose to report the cause of the error using the SOAP fault mechanism. The SAML token profile of
 1103 [WSS: SOAP Message Security](#) does not require that SOAP faults be returned for such errors, and
 1104 systems that choose to return faults SHOULD take care not to introduce any security vulnerabilities as a
 1105 result of the information returned in error responses.

1106 Systems that choose to return faults SHOULD respond with the error codes and fault strings defined in the
 1107 [WSS: SOAP Message Security](#) specification. The RECOMMENDED correspondence between the
 1108 common assertion processing failures and the error codes defined in [WSS: SOAP Message Security](#) are
 1109 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

1110 The preceding table defines fault codes in a form suitable for use with SOAP 1.1. The [WSS: SOAP](#)
 1111 [Message Security](#) specification describes how to map SOAP 1.1 fault constructs to the SOAP 1.2 fault
 1112 constructs.

1113 4 Threat Model and Countermeasures (non- 1114 normative)

1115 This document defines the mechanisms and procedures for securely attaching SAML assertions to SOAP
1116 messages. SOAP messages are used in multiple contexts, specifically including cases where the
1117 message is transported without an active session, the message is persisted, or the message is routed
1118 through a number of intermediaries. Such a general context of use suggests that users of this profile must
1119 be concerned with a variety of threats.

1120 In general, the use of SAML assertions with [WSS: SOAP Message Security](#) introduces no new threats
1121 beyond those identified for SAML or by the [WSS: SOAP Message Security](#) specification. The following
1122 sections provide an overview of the characteristics of the threat model, and the countermeasures that
1123 SHOULD be adopted for each perceived threat.

1124 4.1 Eavesdropping

1125 Eavesdropping is a threat to the SAML token profile of [WSS: SOAP Message Security](#) in the same
1126 manner as it is a threat to any network protocol. The routing of SOAP messages through intermediaries
1127 increases the potential incidences of eavesdropping. Additional opportunities for eavesdropping exist
1128 when SOAP messages are persisted.

1129 To provide maximum protection from eavesdropping, assertions, assertion references, and sensitive
1130 message content SHOULD be encrypted such that only the intended audiences can view their content.
1131 This approach removes threats of eavesdropping in transit, but MAY not remove risks associated with
1132 storage or poor handling by the receiver.

1133 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1134 references from eavesdropping while in transport, but message content MUST be encrypted above the
1135 transport if it is to be protected from eavesdropping by intermediaries.

1136 4.2 Replay

1137 Reliance on authority-protected (e.g., signed) assertions with a holder-of-key subject confirmation
1138 mechanism precludes all but a holder of the key from binding the assertions to a SOAP message.
1139 Although this mechanism effectively restricts data origin to a holder of the confirmation key, it does not, by
1140 itself, provide the means to detect the capture and resubmission of the message by other parties.

1141 Assertions that contain a sender-vouches confirmation mechanism introduce another dimension to replay
1142 vulnerability if the assertions impose no restriction on the entities that may use or reuse the assertions.

1143 Replay attacks can be detected by receivers if message senders include additional message identifying
1144 information (e.g., timestamps, nonces, and or recipient identifiers) within origin-protected message
1145 content and receivers check this information against previously received values.

1146 4.3 Message Insertion

1147 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message insertion attacks.

1148 4.4 Message Deletion

1149 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message deletion attacks.

1150 4.5 Message Modification

1151 Messages constructed according to this specification are protected from message modification if receivers
1152 can detect unauthorized modification of relevant message content. Therefore, it is strongly
1153 RECOMMENDED that all relevant and immutable message content be signed by an attesting entity.
1154 Receivers SHOULD only consider the correspondence between the subject of the SAML assertions and

1155 the SOAP message content to have been established for those portions of the message that are protected
1156 by the attesting entity against modification by another entity.

1157 To ensure that message receivers can have confidence that received assertions have not been forged or
1158 altered since their issuance, SAML assertions appearing in or referenced from `<wsse:Security>`
1159 header elements MUST be protected against unauthorized modification (e.g., signed) by their issuing
1160 authority or the attesting entity (as the case warrants). It is strongly RECOMMENDED that an attesting
1161 entity sign any `<saml:Assertion>` elements that it is attesting for and that are not signed by their
1162 issuing authority.

1163 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1164 assertion references from modification while in transport, but signatures are required to extend such
1165 protection through intermediaries.

1166 **4.6 Man-in-the-Middle**

1167 Assertions with a holder-of-key subject confirmation method are not vulnerable to a MITM attack.
1168 Assertions with a sender-vouches subject confirmation method are vulnerable to MITM attacks to the
1169 degree that the receiver does not have a trusted binding of key to the attesting entity's identity.

5 References

- 1170
- 1171 **[GLOSSARY]** Informational RFC 2828, "[Internet Security Glossary](#)," May 2000.
- 1172 **[KEYWORDS]** S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," [RFC 2119](#), Harvard University, March 1997
- 1173
- 1174 **[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.
- 1175
- 1176
- 1177 **[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.
- 1178
- 1179
- 1180 **[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.
- 1181
- 1182
- 1183 **[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.
- 1184
- 1185
- 1186 **[SOAP]** W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.
- 1187 W3C Working Draft, Nilo Mitra (Editor), [SOAP Version 1.2 Part 0: Primer](#), June 2002.
- 1188
- 1189 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-
- 1190 Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 1: Messaging Framework](#), June 2002.
- 1191
- 1192 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-
- 1193 Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 2: Adjuncts](#), June 2002.
- 1194
- 1195 **[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.
- 1196
- 1197
- 1198 **[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.
- 1199
- 1200 **[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.
- 1201
- 1202 **[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.
- 1203
- 1204
- 1205 **[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.
- 1206
- 1207
- 1208 **[XML-ns]** W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.
- 1209 **[XML Signature]** W3C Recommendation, "[XML Signature Syntax and Processing](#)," 12 February 2002.
- 1210
- 1211 **[XML Token]** Contribution to the WSS TC, Chris Kaler (Editor),
- 1212 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
--------	-----------	--------