



Web Services Security: SAML Token Profile 1.1

OASIS Committee Specification, 14 November 2005

Document Identifier:

[wss-v1.1-spec-cs-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 was last revised or approved by the membership of the Web Services Security TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

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

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

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

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

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

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

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

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

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

Table of Contents

65	1 Introduction.....	4
66	1.1 Goals.....	4
67	1.1.1 Non-Goals.....	4
68	2 Notations and Terminology.....	5
69	2.1 Notational Conventions.....	5
70	2.2 Namespaces.....	5
71	2.3 Terminology.....	6
72	3 Usage.....	7
73	3.1 Processing Model.....	7
74	3.2 SAML Version Differences.....	7
75	3.2.1 Assertion Identifier.....	7
76	3.2.2 Relationship of Subjects to Statements.....	7
77	3.2.3 Assertion URI Reference Replaces AuthorityBinding.....	9
78	3.2.4 Attesting Entity Identifier.....	9
79	3.3 Attaching Security Tokens.....	9
80	3.4 Identifying and Referencing Security Tokens.....	10
81	3.4.1 SAML Assertion Referenced from Header or Element.....	12
82	3.4.2 SAML Assertion Referenced from KeyInfo.....	13
83	3.4.3 SAML Assertion Referenced from SignedInfo.....	15
84	3.4.4 SAML Assertion Referenced from Encrypted Data Reference.....	16
85	3.4.5 SAML Version Support and Backward Compatibility.....	16
86	3.5 Subject Confirmation of SAML Assertions.....	16
87	3.5.1 Holder-of-key Subject Confirmation Method.....	17
88	3.5.2 Sender-vouches Subject Confirmation Method.....	21
89	3.5.3 Bearer Confirmation Method.....	24
90	3.6 Error Codes.....	25
91	4 Threat Model and Countermeasures (non-normative).....	26
92	4.1 Eavesdropping.....	26
93	4.2 Replay.....	26
94	4.3 Message Insertion.....	26
95	4.4 Message Deletion.....	26
96	4.5 Message Modification.....	26
97	4.6 Man-in-the-Middle.....	27
98	5 References	28
99	Appendix B. Acknowledgments.....	29
100		

101 **1 Introduction**

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

106 **1.1 Goals**

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

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

113 **1.1.1 Non-Goals**

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

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

139 **2.3 Terminology**

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

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

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

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

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

151 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.

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

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

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

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



```

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

```

274 3.2.3 Assertion URI Reference Replaces AuthorityBinding

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

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

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

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

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

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

284 3.2.4 Attesting Entity Identifier

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

```

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

```

296 3.3 Attaching Security Tokens

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

```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

403 identifies the type and version of the referenced security token. When the referenced assertion is a SAML
 404 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

405 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

406 Table-3 TokenType Attribute Values

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

410 3.4.1 SAML Assertion Referenced from Header or Element

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

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

```

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

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

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

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

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

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

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

488 3.4.2 SAML Assertion Referenced from KeyInfo

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

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

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

565 3.4.3 SAML Assertion Referenced from SignedInfo

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

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

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

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

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

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

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

627 3.4.4 SAML Assertion Referenced from Encrypted Data Reference

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

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

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

651 3.4.5 SAML Version Support and Backward Compatibility

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

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

661 3.5 Subject Confirmation of SAML Assertions

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

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

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

672 The following table enumerates the mandatory subject confirmation methods and summarizes their
673 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.

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

679 3.5.1 Holder-of-key Subject Confirmation Method

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

683 3.5.1.1 Attesting Entity

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

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

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

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

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

704 3.5.1.2 Receiver

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

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

713 3.5.1.3 Example V1.1

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

```
717 <?xml version="1.0" encoding="UTF-8"?>
718 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
719   <S12:Header>
720
721     <wsse:Security xmlns:wsse="..." xmlns:wsse11="..." xmlns:ds="...">
722       <saml:Assertion xmlns:saml="..."
723         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
724         IssueInstant="2005-05-27T16:53:33.173Z"
725         Issuer="www.opensaml.org"
726         MajorVersion="1"
727         MinorVersion="1">
728         <saml:Conditions
729           NotBefore="2005-05-27T16:53:33.173Z"
730           NotOnOrAfter="2005-05-27T16:58:33.17302Z"/>
731         <saml:AttributeStatement>
732           <saml:Subject>
733             <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)`.

```

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

```

799 **3.5.1.4 Example V2.0**

800 The following example illustrates the use of the holder-of-key subject confirmation method to establish the
 801 correspondence between the SOAP message and the subject of the SAML V2.0 assertion in the
 802 <wsse:Security> header:

```

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

```

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

865 3.5.2 Sender-vouches Subject Confirmation Method

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

869 3.5.2.1 Attesting Entity

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

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

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

884 3.5.2.2 Receiver

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

890 3.5.2.3 Example V1.1

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

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

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

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

```

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

```

```

978         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
979         <ds:DigestValue>...</ds:DigestValue>
980     </ds:Reference>
981 </ds:SignedInfo>
982 <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
983 <ds:KeyInfo>
984     <wsse:SecurityTokenReference wsu:Id="STR2"
985         wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
986 token-profile-1.1#SAMLV1.1">
987         <wsse:KeyIdentifier wsu:Id="..."
988             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
989 profile-1.0#SAMLAssertionID">
990             _a75adf55-01d7-40cc-929f-dbd8372ebdfc
991         </wsse:KeyIdentifier>
992     </wsse:SecurityTokenReference>
993 </ds:KeyInfo>
994 </ds:Signature>
995 </wsse:Security>
996 </S12:Header>
997
998 <S12:Body wsu:Id="MsgBody">
999     <ReportRequest>
1000         <TickerSymbol>SUNW</TickerSymbol>
1001     </ReportRequest>
1002 </S12:Body>
1003 </S12:Envelope>

```

1004 3.5.2.4 Example V2.0

1005 The following example illustrates the mapping of the preceding example to SAML V2.0 assertions.

```

1006 <?xml version="1.0" encoding="UTF-8"?>
1007 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
1008     <S12:Header>
1009
1010         <wsse:Security xmlns:wsse="..." xmlns:wssell="..." xmlns:ds="...">
1011             <saml2:Assertion xmlns:saml2="..." xmlns:xsi="..."
1012
1013                 ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
1014                 <saml2:Subject>
1015                     <saml2:NameID>
1016                         ...
1017                     </saml2:NameID>
1018                 <saml2:SubjectConfirmation
1019                     Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
1020                     <saml2:SubjectConfirmationData
1021                         xsi:type="saml2:KeyInfoConfirmationDataType">
1022                         <ds:KeyInfo>
1023                             <ds:KeyValue>...</ds:KeyValue>
1024                         </ds:KeyInfo>
1025                     </saml2:SubjectConfirmationData>
1026                     </saml2:SubjectConfirmation>
1027                 </saml2:Subject>
1028                 <saml2:Statement>
1029                     ...
1030                 </saml2:Statement>
1031                 <ds:Signature>...</ds:Signature>
1032             </saml2:Assertion>
1033
1034             <wsse:SecurityTokenReference wsu:Id="STR1"
1035                 wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1036 profile-1.1#SAMLV2.0">
1037                 <wsse:Reference wsu:Id="..."
1038                     URI="https://www.opensaml.org?_a75adf55-01d7-40cc-929f-
1039 dbd8372ebdbe">
1040                 </wsse:Reference>

```

```

1041     </wsse:SecurityTokenReference>
1042
1043     <ds:Signature>
1044         <ds:SignedInfo>
1045             <ds:CanonicalizationMethod
1046                 Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1047             <ds:SignatureMethod
1048                 Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
1049             <ds:Reference URI="#STR1">
1050                 <Transforms>
1051                     <ds:Transform
1052
1053                         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
1054 wss-soap-message-security-1.0#STR-Transform">
1055                         <wsse:TransformationParameters>
1056                             <ds:CanonicalizationMethod
1057                                 Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
1058                             </wsse:TransformationParameters>
1059                         </ds:Transform>
1060                     </Transforms>
1061                     <ds:DigestMethod
1062                         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1063                     <ds:DigestValue>...</ds:DigestValue>
1064                 </ds:Reference>
1065                 <ds:Reference URI="#MsgBody">
1066                     <ds:DigestMethod
1067                         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
1068                     <ds:DigestValue>...</ds:DigestValue>
1069                 </ds:Reference>
1070             </ds:SignedInfo>
1071             <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
1072             <ds:KeyInfo>
1073                 <wsse:SecurityTokenReference wsu:Id="STR2">
1074                     wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
1075 token-profile-1.1#SAMLV2.0">
1076                     <wsse:KeyIdentifier wsu:Id="..."
1077                         ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
1078 profile-1.1#SAMLID">
1079                         a75adf55-01d7-40cc-929f-dbd8372ebdfc
1080                     </wsse:KeyIdentifier>
1081                 </wsse:SecurityTokenReference>
1082             </ds:KeyInfo>
1083         </ds:Signature>
1084     </wsse:Security>
1085 </S12:Header>
1086
1087 <S12:Body wsu:Id="MsgBody">
1088     <ReportRequest>
1089         <TickerSymbol>SUNW</TickerSymbol>
1090     </ReportRequest>
1091 </S12:Body>
1092 </S12:Envelope>

```

1093 3.5.3 Bearer Confirmation Method

1094 This profile does NOT require message receivers to establish the relationship between a received
1095 message and the statements of any bearer confirmed (i.e., confirmation method
1096 urn:oasis:names:tc:SAML:1.0:cm:bearer) assertions conveyed or referenced from the message.
1097 Conformant implementations of this profile MUST be able to process references and convey bearer
1098 assertions within <wsse:Security> headers. Any additional processing requirements that pertain
1099 specifically to bearer confirmed assertions are outside the scope of this profile.

1100 **3.6 Error Codes**

1101 When a system that implements the SAML token profile of [WSS: SOAP Message Security](#) does not
 1102 perform its normal processing because of an error detected during the processing of a security header, it
 1103 MAY choose to report the cause of the error using the SOAP fault mechanism. The SAML token profile of
 1104 [WSS: SOAP Message Security](#) does not require that SOAP faults be returned for such errors, and
 1105 systems that choose to return faults SHOULD take care not to introduce any security vulnerabilities as a
 1106 result of the information returned in error responses.

1107 Systems that choose to return faults SHOULD respond with the error codes and fault strings defined in the
 1108 [WSS: SOAP Message Security](#) specification. The RECOMMENDED correspondence between the
 1109 common assertion processing failures and the error codes defined in [WSS: SOAP Message Security](#) are
 1110 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

1111 The preceding table defines fault codes in a form suitable for use with SOAP 1.1. The [WSS: SOAP](#)
 1112 [Message Security](#) specification describes how to map SOAP 1.1 fault constructs to the SOAP 1.2 fault
 1113 constructs.

1114 4 Threat Model and Countermeasures (non- 1115 normative)

1116 This document defines the mechanisms and procedures for securely attaching SAML assertions to SOAP
1117 messages. SOAP messages are used in multiple contexts, specifically including cases where the
1118 message is transported without an active session, the message is persisted, or the message is routed
1119 through a number of intermediaries. Such a general context of use suggests that users of this profile must
1120 be concerned with a variety of threats.

1121 In general, the use of SAML assertions with [WSS: SOAP Message Security](#) introduces no new threats
1122 beyond those identified for SAML or by the [WSS: SOAP Message Security](#) specification. The following
1123 sections provide an overview of the characteristics of the threat model, and the countermeasures that
1124 SHOULD be adopted for each perceived threat.

1125 4.1 Eavesdropping

1126 Eavesdropping is a threat to the SAML token profile of [WSS: SOAP Message Security](#) in the same
1127 manner as it is a threat to any network protocol. The routing of SOAP messages through intermediaries
1128 increases the potential incidences of eavesdropping. Additional opportunities for eavesdropping exist
1129 when SOAP messages are persisted.

1130 To provide maximum protection from eavesdropping, assertions, assertion references, and sensitive
1131 message content SHOULD be encrypted such that only the intended audiences can view their content.
1132 This approach removes threats of eavesdropping in transit, but MAY not remove risks associated with
1133 storage or poor handling by the receiver.

1134 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1135 references from eavesdropping while in transport, but message content MUST be encrypted above the
1136 transport if it is to be protected from eavesdropping by intermediaries.

1137 4.2 Replay

1138 Reliance on authority-protected (e.g., signed) assertions with a holder-of-key subject confirmation
1139 mechanism precludes all but a holder of the key from binding the assertions to a SOAP message.
1140 Although this mechanism effectively restricts data origin to a holder of the confirmation key, it does not, by
1141 itself, provide the means to detect the capture and resubmission of the message by other parties.

1142 Assertions that contain a sender-vouches confirmation mechanism introduce another dimension to replay
1143 vulnerability if the assertions impose no restriction on the entities that may use or reuse the assertions.

1144 Replay attacks can be detected by receivers if message senders include additional message identifying
1145 information (e.g., timestamps, nonces, and or recipient identifiers) within origin-protected message
1146 content and receivers check this information against previously received values.

1147 4.3 Message Insertion

1148 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message insertion attacks.

1149 4.4 Message Deletion

1150 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message deletion attacks.

1151 4.5 Message Modification

1152 Messages constructed according to this specification are protected from message modification if receivers
1153 can detect unauthorized modification of relevant message content. Therefore, it is strongly
1154 RECOMMENDED that all relevant and immutable message content be signed by an attesting entity.
1155 Receivers SHOULD only consider the correspondence between the subject of the SAML assertions and

1156 the SOAP message content to have been established for those portions of the message that are protected
1157 by the attesting entity against modification by another entity.

1158 To ensure that message receivers can have confidence that received assertions have not been forged or
1159 altered since their issuance, SAML assertions appearing in or referenced from `<wsse:Security>`
1160 header elements MUST be protected against unauthorized modification (e.g., signed) by their issuing
1161 authority or the attesting entity (as the case warrants). It is strongly RECOMMENDED that an attesting
1162 entity sign any `<saml:Assertion>` elements that it is attesting for and that are not signed by their
1163 issuing authority.

1164 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1165 assertion references from modification while in transport, but signatures are required to extend such
1166 protection through intermediaries.

1167 **4.6 Man-in-the-Middle**

1168 Assertions with a holder-of-key subject confirmation method are not vulnerable to a MITM attack.
1169 Assertions with a sender-vouches subject confirmation method are vulnerable to MITM attacks to the
1170 degree that the receiver does not have a trusted binding of key to the attesting entity's identity.

5 References

- 1171
- 1172 **[GLOSSARY]** Informational RFC 2828, "[Internet Security Glossary](#)," May 2000.
- 1173 **[KEYWORDS]** S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," [RFC 2119](#), Harvard University, March 1997
- 1174
- 1175 **[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.
- 1176
- 1177
- 1178 **[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.
- 1179
- 1180
- 1181 **[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.
- 1182
- 1183
- 1184 **[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.
- 1185
- 1186
- 1187 **[SOAP]** W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.
- 1188 W3C Working Draft, Nilo Mitra (Editor), [SOAP Version 1.2 Part 0: Primer](#), June 2002.
- 1189
- 1190 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-
- 1191 Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 1: Messaging Framework](#), June 2002.
- 1192
- 1193 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-
- 1194 Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 2: Adjuncts](#), June 2002.
- 1195
- 1196 **[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.
- 1197
- 1198
- 1199 **[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.
- 1200
- 1201 **[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.
- 1202
- 1203 **[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.
- 1204
- 1205
- 1206 **[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.
- 1207
- 1208
- 1209 **[XML-ns]** W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.
- 1210 **[XML Signature]** W3C Recommendation, "[XML Signature Syntax and Processing](#)," 12 February 2002.
- 1211
- 1212 **[XML Token]** Contribution to the WSS TC, Chris Kaler (Editor),
- 1213 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
--------	-----------	--------