



Web Services Security: SAML Token Profile 1.1

Working Draft 04, 13 June 2005

Document Location:

<http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1.pdf>

Errata Location:

<http://www.oasis-open.org/committees/wss>

Technical Committee:

Web Services Security (WSS)

Chairs:

Kelvin Lawrence	IBM
Chris Kaler	Microsoft

Editors:

Ronald Monzillo	Sun
Chris Kaler	Microsoft
Anthony Nadalin	IBM
Phillip Hallam-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.

Status:

This is a working draft. Please send comments to the editors.

Committee members should send comments on this specification to wss@lists.oasis-open.org list. Others should subscribe to and send comments to the wss-comment@lists.oasis-open.org list. To subscribe, visit <http://lists.oasis-open.org/ob/adm.pl>.

32 For information on the disclosure of Intellectual Property Rights or licensing
33 terms related to the work of the Web Services Security TC please refer to the
34 Intellectual Property Rights section of the TC web page at
35 **<http://www.oasis-open.org/committees/wss/>**. The OASIS policy on
36 Intellectual Property Rights is described at
37 **<http://www.oasis-open.org/who/intellectualproperty.shtml>**.
38

38 **Notices**

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

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

53 Copyright © The Organization for the Advancement of Structured Information Standards [OASIS]
54 2002-2005. All Rights Reserved.

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

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

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

72	Table of Contents	
73	1 Introduction.....	6
74	1.1 Goals.....	6
75	1.1.1 Non-Goals	6
76	2 Notations and Terminology	7
77	2.1 Notational Conventions	7
78	2.2 Namespaces	7
79	2.3 Terminology	8
80	3 Usage	9
81	3.1 Processing Model.....	9
82	3.2 SAML Version Differences	9
83	3.2.1 Assertion Identifier	10
84	3.2.2 Relationship of Subjects to Statements.....	10
85	3.2.3 Assertion URI Reference Replaces AuthorityBinding	12
86	3.2.4 Attesting Entity Identifier	12
87	3.3 Attaching Security Tokens	12
88	3.4 Identifying and Referencing Security Tokens	13
89	3.4.1 SAML Assertion Referenced from Header or Element.....	16
90	3.4.2 SAML Assertion Referenced from KeyInfo.....	18
91	3.4.3 SAML Assertion Referenced from SignedInfo	19
92	3.4.4 SAML Assertion Referenced from Encrypted Data Reference	20
93	3.5 Subject Confirmation of SAML Assertions	21
94	3.5.1 Holder-of-key Subject Confirmation Method	22
95	3.5.2 Sender-vouches Subject Confirmation Method	26
96	3.6 Error Codes	31
97	4 Threat Model and Countermeasures (non-normative).....	33
98	4.1 Eavesdropping	33
99	4.2 Replay	33
100	4.3 Message Insertion	34
101	4.4 Message Deletion	34
102	4.5 Message Modification	34
103	4.6 Man-in-the-Middle	34
104	5 References	35
105	Appendix A: Revision History	37

106 Appendix B: Notices39
107

108

1 Introduction

109 The WSS: SOAP Message Security specification defines a standard set of SOAP
110 extensions that implement SOAP message authentication and encryption. This
111 specification defines the use of Security Assertion Markup Language (SAML)
112 assertions as security tokens from the `<wsse:Security>` header block defined by the
113 WSS: SOAP Message Security specification.

1.1 Goals

115 The goal of this specification is to define the use of SAML V1.1 and V2.0 assertions in
116 the context of WSS: SOAP Message Security including for the purpose of securing
117 SOAP messages and SOAP message exchanges. To achieve this goal, this profile
118 describes how:

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

1.1.1 Non-Goals

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

- 124 1. Defining SAML statement syntax or semantics.
- 125 2. Describing the use of SAML assertions other than for SOAP Message Security.
- 126 3. Describing the use of SAML V1.0 assertions with the Web Services Security
127 (WSS): SOAP Message Security specification.

128 2 Notations and Terminology

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

131 2.1 Notational Conventions

132 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
133 "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
134 document are to be interpreted as described in RFC2119.

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

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

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

144 Readers are presumed to be familiar with the terms in the Internet Security
145 Glossary.

146 2.2 Namespaces

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

Prefix	
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-01.xsd
wsse11	TBD

wsu	<code>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd</code>
saml	<code>urn: oasis:names:tc:SAML:1.0:assertion</code>
saml2	<code>urn: oasis:names:tc:SAML:2.0:assertion</code>
samlp	<code>urn: oasis:names:tc:SAML:1.0:protocol</code>

151 **Table-1 Namespace Prefixes**

152 **2.3 Terminology**

153 This specification employs the terminology defined in the WSS: SOAP Message
154 Security specification. The definitions for additional terminology used in this
155 specification appear below.

156

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

160

161 Confirmation Method Identifier – the value within a SAML `SubjectConfirmation`
162 element that identifies the subject confirmation process to be used with the
163 corresponding statements.

164

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

168

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

170

171 Subject – A representation of the entity to which the claims in one or more SAML
172 statements apply.

173 **3 Usage**

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

176 **3.1 Processing Model**

177 This specification extends the token-independent processing model defined by the
178 WSS: SOAP Message Security specification.

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

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

193 **3.2 SAML Version Differences**

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

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.

196 3.2.1 Assertion Identifier

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

200 3.2.2 Relationship of Subjects to Statements

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

207 A SAML V1.1 statement that contains a `<saml:Subject>` element (i.e., a subject
208 statement) may contain a `<saml:SubjectConfirmation>` element that defines the
209 rules for confirming the subject and claims of the statement. If present, the
210 `<saml:SubjectConfirmation>` element occurs within the subject element, and
211 defines one or more methods (i.e., `<saml:ConfirmationMethod>` elements) by which
212 the statement may be confirmed and will include a `<ds:KeyInfo>`³ element when any
213 of the specified methods are based on demonstration of a confirmation key. The
214 `<saml:SubjectConfirmation>` element also provides for the inclusion of additional
215 information to be applied in the confirmation method processing via the optional
216 `<saml:SubjectConfirmationData>` element. The following example depicts a SAML
217 V1.1 assertion containing two subject statements with different subjects and
218 different subject confirmation elements.

```
219 <saml:Assertion  
220   ...  
221   <saml:SubjectStatement>  
222     <saml:Subject>  
223       <saml:NameIdentifier  
224         ...  
225       </saml:NameIdentifier>  
226       <saml:SubjectConfirmation>  
227         <saml:ConfirmationMethod>  
228           urn:oasis:names:tc:SAML:1.0:cm:sender-vouches  
229         </saml:ConfirmationMethod>  
230         <saml:ConfirmationMethod>  
231           urn:oasis:names:tc:SAML:1.0:cm:holder-of-key  
232         </saml:ConfirmationMethod>  
233         <ds:KeyInfo>  
234           <ds:KeyValue>...</ds:KeyValue>  
235         </ds:KeyInfo>  
236       </saml:SubjectConfirmation>  
237     </saml:Subject>  
238     ...  
239   </saml:SubjectStatement>  
240   <saml:SubjectStatement>
```

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

```

241     <saml:Subject>
242         <saml:NameIdentifier
243             ...
244         </saml:NameIdentifier>
245         <saml:SubjectConfirmation>
246             <saml:ConfirmationMethod>
247                 urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
248             </saml:ConfirmationMethod>
249         </saml:SubjectConfirmation>
250     </saml:Subject>
251     ...
252 </saml:SubjectStatement>
253 ...
254 </saml:Assertion>

```

255 A SAML V2.0 assertion may contain a single `<saml2:Subject>` that applies to all the
256 statements of the assertion. When a subject is included in A SAML V2.0 assertion, it
257 may contain any number of `<saml2:SubjectConfirmation>` elements, satisfying any
258 of which is sufficient to confirm the subject and all the statements of the assertion.
259 Each `<saml2:SubjectConfirmation>` element identifies a single confirmation
260 method (by attribute value) and may include an optional
261 `<saml2:SubjectConfirmationData>` element that is used to specify optional
262 confirmation method independent condition attributes and to define additional
263 method specific confirmation data. In the case of a key dependent confirmation
264 method, a `<saml2:KeyInfoConfirmationDataType>` that includes 1 or more
265 `<ds:KeyInfo>` elements is included as `<saml2:SubjectConfirmationData>`. In this
266 case, each `<ds:KeyInfo>` element identifies a key that may be demonstrated to
267 confirm the assertion. The following example depicts a SAML V2.0 assertion
268 containing a subject with multiple confirmation elements that apply to all the
269 statements of the assertion.

```

270 <saml2:Assertion
271     ...
272     <saml2:Subject>
273         <saml2:NameID>
274             ...
275         </saml2:NameID>
276         <saml2:SubjectConfirmation
277             Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
278             <saml2:SubjectConfirmationData
279                 Address="129.148.9.42"
280             </saml2:SubjectConfirmationData>
281         </saml2:SubjectConfirmation>
282         <saml2:SubjectConfirmation
283             Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
284             <saml2:KeyInfoSubjectConfirmationData>
285                 <ds:KeyInfo>
286                     <ds:KeyValue>...</ds:KeyValue>
287                 </ds:KeyInfo>
288             </saml2:KeyInfoSubjectConfirmationData>
289         </saml2:SubjectConfirmation>
290     </saml2:Subject>
291     ...
292 <saml2:Statement>
293     ...
294 </saml2:Statement>
295

```

```
296 <saml2:Statement>
297 ...
298 </saml2:Statement>
299 ...
300
301 </saml2:Assertion>
```

302 3.2.3 Assertion URI Reference Replaces AuthorityBinding

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

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

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

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

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

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

314 3.2.4 Attesting Entity Identifier

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

```
318 <saml2:SubjectConfirmation
319   Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
320   <NameID>
321     gateway
322   </NameID>
323   <saml2:SubjectConfirmationData>
324     Address="129.148.9.42"
325   </saml2:SubjectConfirmationData>
326 </saml2:SubjectConfirmation>
```

327 3.3 Attaching Security Tokens

328 SAML assertions are attached to SOAP messages using WSS: SOAP Message Security
329 by placing assertion elements or references to assertions inside a `<wsse:Security>`
330 header. The following example illustrates a SOAP message containing a bearer
331 confirmed SAML V1.1 assertion in a `<wsse:Security>` header.

```
332 <S12:Envelope>
333   <S12:Header>
334     <wsse:Security>
335
336       <saml:Assertion
337         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
338         IssueInstant="2003-04-17T00:46:02Z"
```

```

339     Issuer="www.opensaml.org"
340     MajorVersion="1"
341     MinorVersion="1"
342     . . .
343     <saml:AuthenticationStatement>
344         <saml:Subject>
345             <saml:NameIdentifier
346                 NameQualifier="www.example.com"
347                 Format="urn:oasis:names:tc:SAML:1.1:nameid-
348 format:X509SubjectName">
349                 uid=joe,ou=people,ou=saml-demo,o=baltimore.com
350             </saml:NameIdentifier>
351             <saml:SubjectConfirmation>
352                 <saml:ConfirmationMethod>
353                     urn:oasis:names:tc:SAML:1.0:cm:bearer
354                 </saml:ConfirmationMethod>
355             </saml:SubjectConfirmation>
356         </saml:Subject>
357     </saml:AuthenticationStatement>
358
359     </saml:Assertion>
360
361     </wsse:Security>
362 </S12:Header>
363 <S12:Body>
364     . . .
365 </S12:Body>
366 </S12:Envelope>

```

367 **3.4 Identifying and Referencing Security Tokens**

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

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

379 When a key identifier is used to reference a SAML assertion, it MUST contain as
380 its element value the corresponding SAML assertion identifier. The key identifier
381 MUST also contain a `ValueType` attribute and the value of this attribute MUST be
382 the value from Table 2 corresponding to the version of the referenced assertion.

383 The key identifier MUST NOT include an `EncodingType`⁴ attribute and the element
384 content of the key identifier MUST be encoded as `xsi:string`.

385 When a key identifier is used to reference a V1.1 SAML assertion that is not
386 contained in the same message as the key identifier, a
387 `<saml:AuthorityBinding>` element MUST be contained in the
388 `<wsse:SecurityTokenReference>` element containing the key identifier. The
389 contents of the `<saml:AuthorityBinding>` element MUST contain values
390 sufficient for the intended recipients of the `<wsse:SecurityTokenReference>` to
391 acquire the identified assertion from the intended Authority. To this end, the
392 value of the `AuthorityKind` attribute of the `<saml:AuthorityBinding>` element
393 MUST be `"samlp:AssertionIdReference"`.

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

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

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

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

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

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

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).

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

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

- 420 • A SAML assertion may be referenced directly from a <wsse:Security> header
 421 element. In this case, the assertion is being conveyed by reference in the
 422 message.
- 423 • A SAML assertion may be referenced from a <ds:KeyInfo> element of a
 424 <ds:Signature> element in a <wsse:Security> header. In this case, the
 425 assertion contains a SubjectConfirmation element that identifies the key used
 426 in the signature calculation.
- 427 • A SAML assertion reference may be referenced from a <ds:Reference> element
 428 within the <ds:SignedInfo> element of a <ds:Signature> element in a
 429 <wsse:Security> header. In this case, the doubly-referenced assertion is signed
 430 by the containing signature.
- 431 • A SAML assertion reference may occur as encrypted content within an
 432 <xenc:EncryptedData> element referenced from a <xenc:DataReference>
 433 element within an <xenc:ReferenceList> element. In this case, the assertion
 434 reference (which may contain an embedded assertion) is encrypted.

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

- 436 • local – in which case, it is included in the <wsse:Security> header containing
 437 the reference.
- 438 • remote – in which case it is not included in the <wsse:Security> header
 439 containing the reference, but may occur in another part of the SOAP message or
 440 may be available at the location identified by the reference which may be an
 441 assertion authority.

442 A SAML key identifier reference MUST be used for all (local and remote) references
 443 to SAML 1.1 assertions. All (local and remote) references to SAML V2.0 assertions
 444 SHOULD be by Direct reference and all remote references to V2.0 assertions MUST
 445 be by Direct reference URI. A key identifier reference MAY be used to reference a
 446 local V2.0 assertion. To maintain compatibility with Web Services Security: SOAP
 447 Message Security 1.0, the practice of referencing local SAML 1.1 assertions by Direct
 448 <wsse:SecurityTokenReference> reference is not defined by this profile.

449 Every key identifier, direct, or embedded reference to a SAML assertion SHOULD
 450 contain a wss11:TokenType attribute and the value of this attribute MUST be the
 451 value from Table 3 that identifies the type and version of the referenced security
 452 token. When the referenced assertion is a SAML V2.0 Assertion the reference MUST
 453 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
------	--

454 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

455 Table-3 TokenType Attribute Values

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

460 3.4.1 SAML Assertion Referenced from Header or Element

461 All conformant implementations MUST be able to process SAML assertion references
 462 occurring in a <wsse:Security> header or in a header element other than a
 463 signature to acquire the corresponding assertion. A conformant implementation
 464 MUST be able to process any such reference independent of the confirmation method
 465 of the referenced assertion.

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

```

470 <S12:Envelope>
471   <S12:Header>
472     <wsse:Security>
473       <saml:Assertion
474         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
475         IssueInstant="2003-04-17T00:46:02Z"
476         Issuer="www.opensaml.org"
477         MajorVersion="1"
478         MinorVersion="1"
479         . . .
480       </saml:Assertion>
481       <wsse:SecurityTokenReference wsu:Id="STR1"
482         wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-
483 saml-token-profile-1.1#SAMLV1.1">
484         <wsse:KeyIdentifier wsu:Id="..."
485           ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-
486 token-profile-1.0#SAMLAssertionID">
487           _a75adf55-01d7-40cc-929f-dbd8372ebdfc
488         </wsse:KeyIdentifier>
489       </wsse:SecurityTokenReference>
490     </wsse:Security>

```



```
491 </S12:Header>
492 <S12:Body>
493   . . .
494 </S12:Body>
495 </S12:Envelope>
```

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

```
498 <wsse:SecurityTokenReference
499   wsu:Id="STR1"
500   wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
501   token-profile-1.1#SAMLV2.0">
502   <wsse:KeyIdentifier wsu:Id="..."
503     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
504   profile-1.1#SAMLID">
505     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
506   </wsse:KeyIdentifier>
507 </wsse:SecurityTokenReference>
```

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

```
513 <wsse:SecurityTokenReference wsu:Id="STR1"
514   wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
515   token-profile-1.1#SAMLV1.1">
516   <saml:AuthorityBinding>
517     Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
518     Location="http://www.opensaml.org/SAML-Authority"
519     AuthorityKind= "samlp:AssertionIdReference"
520   </saml:AuthorityBinding>
521   <wsse:KeyIdentifier
522     wsu:Id="..."
523     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
524   profile-1.0#SAMLAssertionID">
525     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
526   </wsse:KeyIdentifier>
527 </wsse:SecurityTokenReference>
```

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

```
530 </wsse:SecurityTokenReference
531   wsu:Id="..."
532   wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
533   token-profile-1.1#SAMLV2.0">
534   <wsse:Reference
535     wsu:Id="..."
536     URI="https://saml.example.edu/assertion-authority?ID=abcde">
537   </wsse:Reference>
538 </wsse:SecurityTokenReference>
```

539 3.4.2 SAML Assertion Referenced from KeyInfo

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

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

```
545 <ds:KeyInfo>  
546   <wsse:SecurityTokenReference wsu:Id="STR1"  
547     wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-  
548 token-profile-1.1#SAMLV1.1">  
549     <wsse:KeyIdentifier wsu:Id="..."  
550       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-  
551 profile-1.0#SAMLAssertionID">  
552       _a75adf55-01d7-40cc-929f-dbd8372ebdfc  
553     </wsse:KeyIdentifier>  
554   </wsse:SecurityTokenReference>  
555 </ds:KeyInfo>
```

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

```
559 <ds:KeyInfo>  
560   <wsse:SecurityTokenReference wsu:Id="STR1"  
561     wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-  
562 token-profile-1.1#SAMLV2.0">  
563     <wsse:KeyIdentifier wsu:Id="..."  
564       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-  
565 profile-1.0#SAMLID">  
566       _a75adf55-01d7-40cc-929f-dbd8372ebdfc  
567     </wsse:KeyIdentifier>  
568   </wsse:SecurityTokenReference>  
569 </ds:KeyInfo>
```

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

```
574 <ds:KeyInfo>  
575   <wsse:SecurityTokenReference wsu:Id="STR1"  
576     wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-  
577 token-profile-1.1#SAMLV1.1">  
578     <saml:AuthorityBinding>  
579       Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"  
580       Location="http://www.opensaml.org/SAML-Authority"  
581       AuthorityKind="samlp:AssertionIdReference"  
582     </saml:AuthorityBinding>  
583     <wsse:KeyIdentifier wsu:Id="..."  
584       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-  
585 profile-1.0#SAMLAssertionID">  
586       _a75adf55-01d7-40cc-929f-dbd8372ebdfc  
587     </wsse:KeyIdentifier>  
588   </wsse:SecurityTokenReference>  
589 </ds:KeyInfo>
```

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

```
593 <ds:KeyInfo>  
594   <wsse:SecurityTokenReference  
595     wsu:id="STR1"  
596     wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-  
597 token-profile-1.1#SAMLV2.0">  
598     <wsse:Reference  
599       wsu:id="..."  
600       URI="https://saml.example.edu/assertion-authority?ID=abcde">  
601     </wsse:Reference>  
602   </wsse:SecurityTokenReference>  
603 </ds:KeyInfo>
```

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

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

613 **3.4.3 SAML Assertion Referenced from SignedInfo**

614 Independent of the confirmation method of the referenced assertion, all conformant
615 implementations MUST be able to process SAML assertions referenced by
616 <wsse:SecurityTokenReference> from <ds:Reference> elements within the
617 <ds:SignedInfo> element of a <ds:Signature> element in a <wsse:Security>
618 header. Embedded references may be digested directly, thus effectively digesting the
619 encapsulated assertion. Other <wsse:SecurityTokenReference> forms must be
620 dereferenced for the referenced assertion to be digested.

621 The core specification, WSS: SOAP Message Security, defines the STR Dereference
622 transform to cause the replacement (in the digest stream) of a
623 <wsse:SecurityTokenReference> with the contents of the referenced token. The
624 STR Dereference transform MUST be specified and applied to digest any SAML
625 assertion that is referenced by a <wsse:SecurityTokenReference> that is not an
626 embedded reference. The STR Dereference transform SHOULD NOT be applied to an
627 embedded reference.

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

```
631 <wsse:SecurityTokenReference wsu:Id="STR1"  
632   wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-  
633 token-profile-1.1#SAMLV1.1">  
634   <saml:AuthorityBinding>  
635     Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
```

```

636     Location="http://www.opensaml.org/SAML-Authority"
637     AuthorityKind= "samlp:AssertionIdReference"
638 </saml:AuthorityBinding>
639 <wsse:KeyIdentifier wsu:Id="..."
640     ValueType="http://docs.oasis-open.org/wss/oasis-2004XX-wss-saml-
641 token-profile-1.0#SAMLAssertionID">
642     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
643 </wsse:KeyIdentifier>
644 </wsse:SecurityTokenReference>
645     . . .
646 <ds:SignedInfo>
647   <ds:CanonicalizationMethod
648     Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
649   <ds:SignatureMethod
650     Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
651   <ds:Reference URI="#STR1">
652     <Transforms>
653       <ds:Transform
654         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
655 wss-soap-message-security-1.0#STR-Transform"/>
656       <wsse:TransformationParameters>
657         <ds:CanonicalizationMethod
658           Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
659       </wsse:TransformationParameters>
660     </ds:Transform>
661   </Transforms>
662   <ds:DigestMethod
663     Algorithm= "http://www.w3.org/2000/09/xmldsig#sha1"/>
664
665     <ds:DigestValue>...</ds:DigestValue>
666   </ds:Reference>
667 </ds:SignedInfo>

```

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

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

676 **3.4.4 SAML Assertion Referenced from Encrypted Data** 677 **Reference**

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

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

```
692 <xenc:EncryptedData Id="EncryptedSTR1">  
693   <ds:KeyInfo>  
694     . . .  
695   </ds:KeyInfo>  
696   <xenc:CipherData>  
697     <xenc:CipherValue>...</xenc:CipherValue>  
698   </xenc:CipherData>  
699 </xenc:EncryptedData>  
700 <xenc:ReferenceList>  
701   <xenc:DataReference URI="#EncryptedSTR1"/>  
702 </xenc:ReferenceList>
```

703 **3.4.5 SAML Version Support and Backward Compatibility**

704 An implementation of this profile MUST satisfy all of its requirements with respect to
705 either or both SAML V1.1 or SAML V2.0 Assertions. An implementation that satisfies
706 the requirements of this profile with respect to SAML V1.1 assertions MUST be able
707 to fully interoperate with any fully compatible implementation of version 1.0 of this
708 profile.

709 An implementation that does not satisfy the requirements of this profile with respect
710 to SAML V1.1 or SAML V2.0 assertions MUST reject a message containing a
711 <wsse:Security> header that references or conveys an assertion of the unsupported
712 version. When a message containing an unsupported assertion version is detected,
713 the receiver MAY choose to respond with an appropriate fault as defined in Section
714 3.6, "Error Codes".

715 **3.5 Subject Confirmation of SAML Assertions**

716 The SAML profile of WSS: SOAP Message Security requires that systems support the
717 holder-of-key and sender-vouches methods of subject confirmation. It is strongly
718 RECOMMENDED that an XML signature be used to establish the relationship between
719 the message and the statements of the attached assertions. This is especially
720 RECOMMENDED whenever the SOAP message exchange is conducted over an
721 unprotected transport.

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

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

Mechanism	RECOMMENDED Processing Rules
Urn:oasis:names:tc:SAML:1.0:cm:holder-of-key Or urn:oasis:names:tc:SAML:2.0:cm:holder-of-key	The attesting entity demonstrates knowledge of a confirmation key identified in a holder-of-key <code>SubjectConfirmation</code> element within the assertion.
Urn:oasis:names:tc:SAML:1.0:cm:sender-vouches Or urn:oasis:names:tc:SAML:2.0:cm:sender-vouches	The attesting entity, (presumed to be) different from the subject, vouches for the verification of the subject. The receiver 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.

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

736 3.5.1 Holder-of-key Subject Confirmation Method

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

740 3.5.1.1 Attesting Entity

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

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

750 To satisfy the associated confirmation method processing to be performed by the
751 message receiver, the attesting entity MUST demonstrate knowledge of the
752 confirmation key. The attesting entity MAY accomplish this by using the confirmation
753 key to sign content within the message and by including the resulting
754 `<ds:Signature>` element in the `<wsse:Security>` header. `<ds:Signature>`
755 elements produced for this purpose MUST conform to the canonicalization and
756 token pre-pending rules defined in the WSS: SOAP Message Security specification.

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

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

764 **3.5.1.2 Receiver**

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

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

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)`.

776 3.5.1.3 Example V1.1

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

```
780 <?xml version="1.0" encoding="UTF-8"?>
781 <S12:Envelope>
782   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
783   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
784   <S12:Header>
785
786     <wsse:Security>
787       <saml:Assertion
788         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
789         IssueInstant="2005-05-27T16:53:33.173Z"
790         Issuer="www.opensaml.org"
791         MajorVersion="1"
792         MinorVersion="1"
793         xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
794         <saml:Conditions
795           NotBefore="2005-05-27T16:53:33.173Z"
796           NotOnOrAfter="2005-05-27T16:58:33.17302Z"/>
797         <saml:AttributeStatement>
798           <saml:Subject>
799             <saml:NameIdentifier
800               NameQualifier="www.example.com"
801               Format="urn:oasis:names:tc:SAML:1.1:nameid-
802 format:X509SubjectName">
803               uid=joe,ou=people,ou=saml-demo,o=baltimore.com
804             </saml:NameIdentifier>
805             <saml:SubjectConfirmation>
806               <saml:ConfirmationMethod>
807                 urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
808               </saml:ConfirmationMethod>
809               <ds:KeyInfo>
810                 <ds:KeyValue>...</ds:KeyValue>
811               </ds:KeyInfo>
812             </saml:SubjectConfirmation>
813           </saml:Subject>
814           <saml:Attribute
815             AttributeName="MemberLevel"
816             AttributeNamespace="http://www.oasis-
817 open.org/Catalyst2002/attributes">
818             <saml:AttributeValue>gold</saml:AttributeValue>
819           </saml:Attribute>
820           <saml:Attribute
821             AttributeName="E-mail"
822             AttributeNamespace="http://www.oasis-
823 open.org/Catalyst2002/attributes">
824             <saml:AttributeValue>joe@yahoo.com</saml:AttributeValue>
825           </saml:Attribute>
826         </saml:AttributeStatement>
827         <ds:Signature>...</ds:Signature>
828       </saml: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-
847 saml-token-profile-1.1#SAMLV1.1">
848       <wsse:KeyIdentifier wsu:Id="..."
849         ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-
850 token-profile-1.0#SAMLAssertionID">
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 3.5.1.4 Example V2.0

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

```

869 <?xml version="1.0" encoding="UTF-8"?>
870 <S12:Envelope>
871   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
872   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
873   <S12:Header>
874
875     <wsse:Security>
876       <saml2:Assertion
877         ...
878         ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
879         ...>
880       <saml2:subject>
881         <saml2:NameID>
882           ...
883         </saml2:NameID>
884         <saml2:SubjectConfirmation
885           Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
886           <saml2:KeyInfoSubjectConfirmationData>
887             <ds:KeyInfo>

```

```

888         <ds:KeyValue>...</ds:KeyValue>
889         </ds:KeyInfo>
890     </saml2:KeyInfoSubjectConfirmationData>
891     <saml2:SubjectConfirmation>
892 </saml2:Subject>
893 <saml2:Statement>
894     ...
895 </saml2:Statement>
896     <ds:Signature>...</ds:Signature>
897 </saml2:Assertion>
898
899 <ds:Signature>
900     <ds:SignedInfo>
901         <ds:CanonicalizationMethod
902             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
903         <ds:SignatureMethod
904             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
905         <ds:Reference
906             URI="#MsgBody">
907             <ds:DigestMethod
908                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
909             <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
910             </ds:Reference>
911         </ds:SignedInfo>
912         <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
913         <ds:KeyInfo>
914             <wsse:SecurityTokenReference wsu:Id="STR1"
915                 wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-
916 saml-token-profile-1.1#SAMLV2.0">
917                 <wsse:KeyIdentifier wsu:Id="..."
918                     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-
919 token-profile-1.1#SAMLID">
920                     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
921                 </wsse:KeyIdentifier>
922             </wsse:SecurityTokenReference>
923         </ds:KeyInfo>
924     </ds:Signature>
925 </wsse:Security>
926 </S12:Header>
927
928 <S12:Body wsu:Id="MsgBody">
929     <ReportRequest>
930         <TickerSymbol>SUNW</TickerSymbol>
931     </ReportRequest>
932 </S12:Body>
933 </S12:Envelope>

```

934 3.5.2 Sender-vouches Subject Confirmation Method

935 The following sections describe the sender-vouches method of establishing the
936 correspondence between a SOAP message and the SAML assertions added to the
937 SOAP message according to the SAML profile of WSS: SOAP Message Security.

938 3.5.2.1 Attesting Entity

939 An attesting entity uses the sender-vouches confirmation method to assert that it is
940 acting on behalf of the subject of SAML statements attributed with a sender-vouches

941 `SubjectConfirmation` element. Statements attested for by the sender-vouches
942 method MUST be associated, within their containing assertion, with one or more
943 sender-vouches `SubjectConfirmation` elements.

944 To satisfy the associated confirmation method processing of the receiver, the
945 attesting entity MUST protect the vouched for SOAP message content such that the
946 receiver can determine when it has been altered by another party. The attesting
947 entity MUST also cause the vouched for statements (as necessary) and their binding
948 to the message contents to be protected such that unauthorized modification can be
949 detected. The attesting entity MAY satisfy these requirements by including in the
950 corresponding `<wsse:Security>` header a `<ds:Signature>` element that it prepares
951 by using its key to sign the relevant message content and assertions. As defined by
952 the XML Signature specification, the attesting entity MAY identify its key by including
953 a `<ds:KeyInfo>` element within the `<ds:Signature>` element.

954 A `<ds:Signature>` element produced for this purpose MUST conform to the
955 canonicalization and token pre-pending rules defined in the WSS: SOAP Message
956 Security specification.

957 **3.5.2.2 Receiver**

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

964 **3.5.2.3 Example V1.1**

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

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

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

```
979 <?xml version="1.0" encoding="UTF-8"?>  
980 <S12:Envelope>  
981   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
982   xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
983     <S12:Header>
```

```

984 <wsse:Security>
985
986     <saml:Assertion
987       AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
988       IssueInstant="2005-05-27T16:53:33.173Z"
989       Issuer="www.opensaml.org"
990       MajorVersion="1"
991       MinorVersion="1"
992       xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
993     <saml:Conditions>
994       NotBefore="2005-05-27T16:53:33.173Z"
995       NotOnOrAfter="2005-05-27T16:58:33.173Z"/>
996     <saml:AttributeStatement>
997       <saml:Subject>
998         <saml:NameIdentifier
999           NameQualifier="www.example.com"
1000           Format="urn:oasis:names:tc:SAML:1.1:nameid-
1001 format:X509SubjectName">
1002           uid=proxy,ou=system,ou=saml-demo,o=baltimore.com
1003         </saml:NameIdentifier>
1004         <saml:SubjectConfirmation>
1005           <saml:ConfirmationMethod>
1006             urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
1007           </saml:ConfirmationMethod>
1008           <ds:KeyInfo>
1009             <ds:KeyValue>...</ds:KeyValue>
1010           </ds:KeyInfo>
1011         </saml:SubjectConfirmation>
1012       </saml:Subject>
1013       <saml:Attribute
1014         . . .
1015       </saml:Attribute>
1016       . . .
1017     </saml:AttributeStatement>
1018   </saml:Assertion>
1019
1020   <wsse:SecurityTokenReference wsu:Id="STR1">
1021     wsse:TokenType="http://docs.oasis-open.org/wss/oasis-wss-
1022 saml-token-profile-1.1#SAMLV1.1">
1023       <saml:AuthorityBinding>
1024         Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
1025         Location="http://www.opensaml.org/SAML-Authority"
1026         AuthorityKind="samlp:AssertionIdReference"
1027       </saml:AuthorityBinding>
1028     <wsse:KeyIdentifier wsu:Id="..."
1029       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-
1030 token-profile-1.0#SAMLAssertionID">
1031       _a75adf55-01d7-40cc-929f-dbd8372ebdbe
1032     </wsse:KeyIdentifier>
1033   </wsse:SecurityTokenReference>
1034
1035   <ds:Signature>
1036     <ds:SignedInfo>
1037       <ds:CanonicalizationMethod
1038         Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
1039       <ds:SignatureMethod
1040         Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
1041       <ds:Reference URI="#STR1">
1042         <Transforms>
1043           <ds:Transform

```

```

1043         Algorithm="http://docs.oasis-
1044 open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#STR-
1045 Transform"/>
1046         <wsse:TransformationParameters>
1047             <ds:CanonicalizationMethod
1048                 Algorithm="http://www.w3.org/2001/10/xml-exc-
1049 c14n#"/>
1050             </wsse:TransformationParameters>
1051         </ds:Transform>
1052     </Transforms>
1053     <ds:DigestMethod
1054         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
1055     <ds:DigestValue>...</ds:DigestValue>
1056 </ds:Reference>
1057 <ds:Reference URI="#MsgBody">
1058     <ds:DigestMethod
1059         Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
1060     <ds:DigestValue>...</ds:DigestValue>
1061 </ds:Reference>
1062 </ds:SignedInfo>
1063 <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
1064 <ds:KeyInfo>
1065     <wsse:SecurityTokenReference wsu:Id="STR2"
1066         wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-
1067 saml-token-profile-1.1#SAMLV1.1">
1068         <wsse:KeyIdentifier wsu:Id="..."
1069             ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-
1070 token-profile-1.0#SAMLAssertionID">
1071             _a75adf55-01d7-40cc-929f-dbd8372ebdfc
1072         </wsse:KeyIdentifier>
1073     </wsse:SecurityTokenReference>
1074 </ds:KeyInfo>
1075 </ds:Signature>
1076 </wsse:Security>
1077 </S12:Header>
1078
1079 <S12:Body wsu:Id="MsgBody">
1080     <ReportRequest>
1081         <TickerSymbol>SUNW</TickerSymbol>
1082     </ReportRequest>
1083 </S12:Body>
1084 </S12:Envelope>

```

1085 3.5.2.4 Example V2.0

1086 The following example illustrates the mapping of the preceding example to SAML
1087 V2.0 assertions.

```

1088 <?xml version="1.0" encoding="UTF-8"?>
1089 <S12:Envelope>
1090     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1091     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
1092     <S12:Header>
1093
1094         <wsse:Security>
1095             <saml2:Assertion
1096                 ...
1097                 ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
1098                 ...>

```

```

1099     <saml2:subject>
1100         <saml2:NameID>
1101             ...
1102         </saml2:NameID>
1103         <saml2:SubjectConfirmation
1104             Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
1105             <saml2:KeyInfoSubjectConfirmationData>
1106                 <ds:KeyInfo>
1107                     <ds:KeyValue>...</ds:KeyValue>
1108                 </ds:KeyInfo>
1109             </saml2:KeyInfoSubjectConfirmationData>
1110         </saml2:SubjectConfirmation>
1111     </saml2:Subject>
1112     <saml2:Statement>
1113         ...
1114     </saml2:Statement>
1115     <ds:Signature>...</ds:Signature>
1116 </saml2:Assertion>

1117
1118     <wsse:SecurityTokenReference wsu:Id="STR1"
1119         wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-
1120 saml-token-profile-1.1#SAMLV2.0">
1121         <wsse:Reference wsu:Id="..."
1122             URI="https://www.opensaml.org?_a75adf55-01d7-40cc-929f-
1123 dbd8372ebdbe">
1124         </wsse:Reference>
1125     </wsse:SecurityTokenReference>

1126
1127     <ds:Signature>
1128         <ds:SignedInfo>
1129             <ds:CanonicalizationMethod
1130                 Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
1131             <ds:SignatureMethod
1132                 Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
1133             <ds:Reference URI="#STR1">
1134                 <Transforms>
1135                     <ds:Transform
1136
1137                         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-
1138 200401-wss-soap-message-security-1.0#STR-Transform"/>
1139                     <wsse:TransformationParameters>
1140                         <ds:CanonicalizationMethod
1141                             Algorithm="http://www.w3.org/2001/10/xml-exc-
1142 c14n#"/>
1143                         </wsse:TransformationParameters>
1144                     </ds:Transform>
1145                 </Transforms>
1146                 <ds:DigestMethod
1147                     Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
1148                 <ds:DigestValue>...</ds:DigestValue>
1149             </ds:Reference>
1150             <ds:Reference URI="#MsgBody">
1151                 <ds:DigestMethod
1152                     Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
1153                 <ds:DigestValue>...</ds:DigestValue>
1154             </ds:Reference>
1155         </ds:SignedInfo>
1156         <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
1157         <ds:KeyInfo>

```

1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177

```
<wsse:SecurityTokenReference wsu:Id="STR2"  
    wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-  
saml-token-profile-1.1#SAMLV2.0">  
    <wsse:KeyIdentifier wsu:Id="..."  
        ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-  
token-profile-1.1#SAMLID">  
        _a75adf55-01d7-40cc-929f-dbd8372ebdfc  
    </wsse:KeyIdentifier>  
</wsse:SecurityTokenReference>  
</ds:KeyInfo>  
</ds:Signature>  
</wsse:Security>  
</S12:Header>  
  
<S12:Body wsu:Id="MsgBody">  
    <ReportRequest>  
        <TickerSymbol>SUNW</TickerSymbol>  
    </ReportRequest>  
</S12:Body>  
</S12:Envelope>
```

1178 **3.5.3 Bearer Confirmation Method**

1179 This profile does NOT require message receivers to establish the relationship
1180 between a received message and the statements of any bearer confirmed (i.e.,
1181 confirmation method urn:oasis:names:tc:SAML:1.0:cm:bearer) assertions
1182 conveyed or referenced from the message. Conformant implementations of this
1183 profile MUST be able to process references and convey bearer assertions within
1184 <wsse:Security> headers. Any additional processing requirements that pertain
1185 specifically to bearer confirmed assertions are outside the scope of this profile.

1186 **3.6 Error Codes**

1187 When a system that implements the SAML token profile of WSS: SOAP Message
1188 Security does not perform its normal processing because of an error detected during
1189 the processing of a security header, it MAY choose to report the cause of the error
1190 using the SOAP fault mechanism. The SAML token profile of WSS: SOAP Message
1191 Security does not require that SOAP faults be returned for such errors, and systems
1192 that choose to return faults SHOULD take care not to introduce any security
1193 vulnerabilities as a result of the information returned in error responses.

1194 Systems that choose to return faults SHOULD respond with the error codes and fault
1195 strings defined in the WSS: SOAP Message Security specification. The
1196 RECOMMENDED correspondence between the common assertion processing failures
1197 and the error codes defined in WSS: SOAP Message Security are defined in the
1198 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

1199 The preceding table defines fault codes in a form suitable for use with SOAP 1.1. The
1200 WSS: SOAP Message Security specification describes how to map SOAP 1.1 fault
1201 constructs to the SOAP 1.2 fault constructs.

1202

4 Threat Model and Countermeasures (non-normative)

1203

1204 This document defines the mechanisms and procedures for securely attaching SAML
1205 assertions to SOAP messages. SOAP messages are used in multiple contexts,
1206 specifically including cases where the message is transported without an active
1207 session, the message is persisted, or the message is routed through a number of
1208 intermediaries. Such a general context of use suggests that users of this profile must
1209 be concerned with a variety of threats.

1210 In general, the use of SAML assertions with WSS: SOAP Message Security introduces
1211 no new threats beyond those identified for SAML or by the WSS: SOAP Message
1212 Security specification. The following sections provide an overview of the
1213 characteristics of the threat model, and the countermeasures that SHOULD be
1214 adopted for each perceived threat.

1215 4.1 Eavesdropping

1216 Eavesdropping is a threat to the SAML token profile of WSS: SOAP Message Security
1217 in the same manner as it is a threat to any network protocol. The routing of SOAP
1218 messages through intermediaries increases the potential incidences of
1219 eavesdropping. Additional opportunities for eavesdropping exist when SOAP
1220 messages are persisted.

1221 To provide maximum protection from eavesdropping, assertions, assertion
1222 references, and sensitive message content SHOULD be encrypted such that only the
1223 intended audiences can view their content. This approach removes threats of
1224 eavesdropping in transit, but MAY not remove risks associated with storage or poor
1225 handling by the receiver.

1226 Transport-layer security MAY be used to protect the message and contained SAML
1227 assertions and/or references from eavesdropping while in transport, but message
1228 content MUST be encrypted above the transport if it is to be protected from
1229 eavesdropping by intermediaries.

1230 4.2 Replay

1231 Reliance on authority-protected (e.g., signed) assertions with a holder-of-key subject
1232 confirmation mechanism precludes all but a holder of the key from binding the
1233 assertions to a SOAP message. Although this mechanism effectively restricts data
1234 origin to a holder of the confirmation key, it does not, by itself, provide the means to
1235 detect the capture and resubmission of the message by other parties.

1236 Assertions that contain a sender-vouches confirmation mechanism introduce another
1237 dimension to replay vulnerability if the assertions impose no restriction on the
1238 entities that may use or reuse the assertions.

1239 Replay attacks can be detected by receivers if message senders include additional
1240 message identifying information (e.g., timestamps, nonces, and or recipient
1241 identifiers) within origin-protected message content and receivers check this
1242 information against previously received values.

1243 **4.3 Message Insertion**

1244 The SAML token profile of WSS: SOAP Message Security is not vulnerable to
1245 message insertion attacks.

1246 **4.4 Message Deletion**

1247 The SAML token profile of WSS: SOAP Message Security is not vulnerable to
1248 message deletion attacks.

1249 **4.5 Message Modification**

1250 Messages constructed according to this specification are protected from message
1251 modification if receivers can detect unauthorized modification of relevant message
1252 content. Therefore, it is strongly RECOMMENDED that all relevant and immutable
1253 message content be signed by an attesting entity. Receivers SHOULD only consider
1254 the correspondence between the subject of the SAML assertions and the SOAP
1255 message content to have been established for those portions of the message that are
1256 protected by the attesting entity against modification by another entity.

1257 To ensure that message receivers can have confidence that received assertions have
1258 not been forged or altered since their issuance, SAML assertions appearing in or
1259 referenced from `<wsse:Security>` header elements MUST be protected against
1260 unauthorized modification (e.g., signed) by their issuing authority or the attesting
1261 entity (as the case warrants). It is strongly RECOMMENDED that an attesting entity
1262 sign any `<saml:Assertion>` elements that it is attesting for and that are not signed
1263 by their issuing authority.

1264 Transport-layer security MAY be used to protect the message and contained SAML
1265 assertions and/or assertion references from modification while in transport, but
1266 signatures are required to extend such protection through intermediaries.

1267 **4.6 Man-in-the-Middle**

1268 Assertions with a holder-of-key subject confirmation method are not vulnerable to a
1269 MITM attack. Assertions with a sender-vouches subject confirmation method are
1270 vulnerable to MITM attacks to the degree that the receiver does not have a trusted
1271 binding of key to the attesting entity's identity.

1272

5 References

- 1273 [GLOSSARY] Informational RFC 2828, "*Internet Security Glossary*," May
1274 2000.
- 1275 [KEYWORDS] S. Bradner, "Key words for use in RFCs to Indicate Requirement
1276 Levels," *RFC 2119*, Harvard University, March 1997
- 1277 [SAMLBindV1] Oasis Standard, E. Maler, P.Mishra, and R. Philpott (Editors),
1278 *Bindings and Profiles for the OASIS Security Assertion Markup
1279 Language (SAML) V1.1*, September 2003.
- 1280 [SAMLBindV2] Oasis Standard, S. Cantor, F. Hirsch, J. Kemp, R. Philpott, E.
1281 Maler (Editors), *Bindings for the OASIS Security Assertion
1282 Markup Language (SAML) V2.0*, March 2005.
- 1283 [SAMLCoreV1] Oasis Standard, E. Maler, P.Mishra, and R. Philpott (Editors),
1284 *Assertions and Protocols for the OASIS Security Assertion
1285 Markup Language (SAML) V1.1*, September 2003.
- 1286 [SAMLCoreV2] Oasis Standard, S. Cantor, J. Kemp, R. Philpott, E. Maler
1287 (Editors), *Assertions and Protocol for the OASIS Security
1288 Assertion Markup Language (SAML) V2.0*, March 2005.
- 1289 [SOAP] W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May
1290 2000.
- 1291 W3C Working Draft, Nilo Mitra (Editor), *SOAP Version 1.2 Part
1292 0: Primer*, June 2002.
- 1293 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah
1294 Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen
1295 (Editors), *SOAP Version 1.2 Part 1: Messaging Framework*, June
1296 2002.
- 1297 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah
1298 Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen
1299 (Editors), *SOAP Version 1.2 Part 2: Adjuncts*, June 2002.
- 1300 [URI] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource
1301 Identifiers (URI): Generic Syntax," *RFC 2396*, MIT/LCS, U.C.
1302 Irvine, Xerox Corporation, August 1998.
- 1303 [WS-SAML] Contribution to the WSS TC, P. Mishra (Editor), WS-Security
1304 Profile of the Security Assertion Markup Language (SAML)
1305 Working Draft 04, Sept 2002.
- 1306 [WSS: SAML Token Profile] Oasis Standard, P. Hallem-Baker, A. Nadalin, C.
1307 Kaler, R. Monzillo (Editors), *Web Services Security: SAML
1308 Token Profile 1.0*, December 2004.

1309	[WSS: SOAP Message Security]	Oasis Standard, A. Nadalin, C.Kaler, P.
1310		Hallem-Baker, R. Monzillo (Editors), Web Services Security:
1311		SOAP Message Security 1.0 (WS-Security 2004), August 2003.
1312	[XML-ns]	W3C Recommendation, "Namespaces in XML," 14 January
1313		1999.
1314	[XML Signature]	W3C Recommendation, "XML Signature Syntax and
1315		Processing," 12 February 2002.
1316	[XML Token]	Contribution to the WSS TC, Chris Kaler (Editor),
1317		WS-Security Profile for XML-based Tokens, August 2002.

1318

Appendix A: Acknowledgements

1319	Maneesh Sahu	Actional Corp
1320	Gene Thurston	AmberPoint
1321	Frank Siebenlist	Argonne National Laboratory
1322	Hal Lockhart	BEA Systems, Inc.
1323	Corinna Witt	BEA Systems, Inc.
1324	Steve Anderson	BMC Software
1325	Davanum Srinivas	Computer Associates
1326	Rich Levinson	Computer Associates
1327	Thomas DeMartini	ContentGuard
1328	Guillermo Lao	ContentGuard
1329	Merlin Hughes	Cybertrust
1330	Rich Salz	DataPower
1331	Sam Wei	Documentum
1332	Tim Moses	Entrust
1333	Carolina Canales-Valenzuela	Ericsson
1334	Dana Kaufman	Forum Systems, Inc.
1335	Toshihiro Nishimura	Fujitsu
1336	Kefeng Chen	GeoTrust
1337	Irving Reid	Hewlett-Packard
1338	Kojiro Nakayama	Hitachi
1339	Paula Austel	IBM
1340	Derek Fu	IBM
1341	Maryann Hondo	IBM
1342	Kelvin Lawrence	IBM
1343	Hiroshi Maruyama	IBM
1344	Michael McIntosh	IBM
1345	Anthony Nadalin	IBM
1346	Nataraj Nagaratnam	IBM
1347	Ron Williams	IBM
1348	Don Flinn	Individual
1349	Jerry Schwarz	Individual
1350	Bob Morgan	Internet2
1351	Kate Cherry	Lockheed Martin
1352	Paul Cotton	Microsoft Corporation
1353	Vijay Gajjala	Microsoft Corporation
1354	Alan Geller	Microsoft Corporation
1355	Chris Kaler	Microsoft Corporation
1356	Jeff Hodges	Neustar
1357	Frederick Hirsch	Nokia
1358	Senthil Sengodan	Nokia
1359	Abbie Barbir	Nortel Networks
1360	Lloyd Burch	Novell
1361	Charles Knouse	Oblix
1362	Vamsi Motukuru	Oracle
1363	Ramana Turlapati	Oracle
1364	Prateek Mishra	Principal Identity
1365	Andrew Nash	Reactivity

1366	Ben Hammond	RSA Security
1367	Rob Philpott	RSA Security
1368	Martijn de Boer	SAP
1369	Blake Dournaee	Sarvega
1370	Coumara Radja	Sarvega
1371	Pete Wenzel	SeeBeyond Technology Corporation
1372	Manveen Kaur	Sun Microsystems
1373	Eve Maler	Sun Microsystems
1374	Ronald Monzillo	Sun Microsystems
1375	Jan Alexander	Systinet
1376	Symon Chang	Tibco
1377	J Weiland	US Dept of the Navy
1378	Hans Granqvist	VeriSign
1379	Phillip Hallam-Baker	Verisign
1380	Hemma Prafullchandra	VeriSign

Appendix B: Revision History

Rev	Date	What
00	07-Oct-04	Initial draft produced from cd-03 of version 1.0 of the profile. Version 1.1 was created to add support for SAML V2.0 Assertions.
01	19-Jan-05	Expert group draft submitted to TC.
02	17-May-2005	<ol style="list-style-type: none"> 1. Designated as V1.1 profile. 2. Incorporated resolution to issue 250 (which created the <code>TokenType</code> attribute). 3. Began transition of compatibility requirements to allow an implementation to support V1.1, V2.0, or both versions of SAML assertions. 4. Added footnote to clarify processing of bearer confirmation mechanism, and also depicted use of bearer in example.
03	31-May-2005	<ol style="list-style-type: none"> 1. Applied Version 1.0 Errata 2. Applied comments from review. 3. Added section on version support and backward compatibility. 4. Clarified requirements with respect to bearer confirmed assertions.
04	13-June-2005	<ol style="list-style-type: none"> 1. Applied revised document template. 2. Updated contributor list (in Acknowledgements)