



Web Services Security SAML Token Profile Version 1.1.1

Candidate OASIS Standard 01

15 December 2011

Specification URIs

This version:

<http://docs.oasis-open.org/wss-m/wss/v1.1.1/cos01/wss-SAMLTOKENProfile-v1.1.1-cos01.doc>
(Authoritative)
<http://docs.oasis-open.org/wss-m/wss/v1.1.1/cos01/wss-SAMLTOKENProfile-v1.1.1-cos01.html>
<http://docs.oasis-open.org/wss-m/wss/v1.1.1/cos01/wss-SAMLTOKENProfile-v1.1.1-cos01.pdf>

Previous version:

<http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-SAMLTOKENProfile-v1.1.1-csd01.doc>
(Authoritative)
<http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-SAMLTOKENProfile-v1.1.1-csd01.html>
<http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-SAMLTOKENProfile-v1.1.1-csd01.pdf>

Latest version:

<http://docs.oasis-open.org/wss-m/wss/v1.1.1/wss-SAMLTOKENProfile-v1.1.1.doc> (Authoritative)
<http://docs.oasis-open.org/wss-m/wss/v1.1.1/wss-SAMLTOKENProfile-v1.1.1.html>
<http://docs.oasis-open.org/wss-m/wss/v1.1.1/wss-SAMLTOKENProfile-v1.1.1.pdf>

Technical Committee:

OASIS Web Services Security Maintenance (WSS-M) TC

Chair:

David Turner (david.turner@microsoft.com), Microsoft

Editors:

Ronald Monzillo (ronald.monzillo@sun.com), Sun Microsystems
Chris Kaler (ckaler@microsoft.com), Microsoft
Anthony Nadalin (droidsecure@us.ibm.com), IBM
Phillip Hallam-Baker (pbaker@verisign.com), Verisign
Carlo Milono (cmilono@tibco.com), Tibco

Additional artifacts:

This prose specification is one component of a multi-part Work Product which includes:

- [Web Services Security Kerberos Token Profile Version 1.1.1](#)
- [Web Services Security Rights Expression Language \(REL\) Token Profile Version 1.1.1](#)
- [Web Services Security SAML Token Profile Version 1.1.1](#) (this document)
- [Web Services Security: SOAP Message Security Version 1.1.1](#)
- [Web Services Security SOAP Message with Attachments \(SwA\) Profile Version 1.1.1](#)
- [Web Services Security Username Token Profile Version 1.1.1](#)
- [Web Services Security X.509 Certificate Token Profile Version 1.1.1](#)
- XML schemas: <http://docs.oasis-open.org/wss-m/wss/v1.1.1/cos01/xsd/>

Related work:

This specification supersedes:

- *Web Services Security SAML Token Profile 1.1*. 01 November 2006. OASIS Standard incorporating Approved Errata. <http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-errata-os-SAMLTOKENProfile.html>
- *Web Services Security SAML Token Profile 1.1*. 01 November 2006. OASIS Approved Errata. <http://docs.oasis-open.org/wss/v1.1/wss-v1.1-errata-os-SAMLTOKENProfile.html>

Abstract:

This document describes how to use Security Assertion Markup Language (SAML) V1.1 and V2.0 assertions with the *Web Services Security SOAP Message Security Version 1.1.1* 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.

This document integrates specific error corrections or editorial changes to the preceding specification, within the scope of the *Web Services Security* and this TC.

This document introduces a third digit in the numbering convention where the third digit represents a consolidation of error corrections, bug fixes or editorial formatting changes (e.g., 1.1.1); it does not add any new features beyond those of the base specifications (e.g., 1.1).

Status:

This document was last revised or approved by the OASIS Web Services Security Maintenance (WSS-M) TC on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document.

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

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 (<http://www.oasis-open.org/committees/wss-m/ipr.php>).

Citation Format:

When referencing this specification the following citation format should be used:

[WSS-SAML-Token-Profile-V1.1.1]

Web Services Security SAML Token Profile Version 1.1.1. 15 December 2011. Candidate OASIS Standard 01. <http://docs.oasis-open.org/wss-m/wss/v1.1.1/cos01/wss-SAMLTOKENProfile-v1.1.1-cos01.html>.

Notices

Copyright © OASIS Open 2012. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](#) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

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

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

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for above guidance.

Table of Contents

1	Introduction.....	5
1.1	Goals.....	5
1.1.1	Non-Goals	5
2	Notations and Terminology	6
2.1	Notational Conventions.....	6
2.2	Namespaces.....	6
2.3	Terminology	7
3	Usage	8
3.1	Processing Model	8
3.2	SAML Version Differences.....	8
3.2.1	Assertion Identifier.....	8
3.2.2	Relationship of Subjects to Statements	8
3.2.3	Assertion URI Reference Replaces AuthorityBinding	11
3.2.4	Attesting Entity Identifier.....	11
3.3	Attaching Security Tokens	11
3.4	Identifying and Referencing Security Tokens.....	12
3.4.1	SAML Assertion Referenced from Header or Element	14
3.4.2	SAML Assertion Referenced from KeyInfo	16
3.4.3	SAML Assertion Referenced from SignedInfo	18
3.4.4	SAML Assertion Referenced from Encrypted Data Reference	19
3.4.5	SAML Version Support and Backward Compatibility	20
3.5	Subject Confirmation of SAML Assertions.....	20
3.5.1	Holder-of-key Subject Confirmation Method	21
3.5.2	Sender-vouches Subject Confirmation Method.....	25
3.5.3	Bearer Confirmation Method	30
3.6	Error Codes.....	30
4	Threat Model and Countermeasures (non-normative)	32
4.1	Eavesdropping.....	32
4.2	Replay.....	32
4.3	Message Insertion	32
4.4	Message Deletion	32
4.5	Message Modification	32
4.6	Man-in-the-Middle	33
5	References	34
6	Conformance	35
A.	Acknowledgements	36
B.	Revision History.....	39

1 Introduction

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

6 1.1 Goals

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

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

13 1.1.1 Non-Goals

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

- 15 1. Defining SAML statement syntax or semantics.
- 16 2. Describing the use of SAML assertions other than for SOAP Message Security.
- 17 1. Describing the use of SAML V1.0 assertions with the [Web Services Security \(WSS\): SOAP Message](#)
18 [Security](#) specification.

19 2 Notations and Terminology

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

21 2.1 Notational Conventions

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

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

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

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

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

34 2.2 Namespaces

35 The appearance of the following [XML-ns] namespace prefixes in the examples within this specification
36 should be understood to refer to the corresponding namespaces (from the following table) whether or
37 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	<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>
xsi	http://www.w3.org/2001/XMLSchema-instance

38 Table-1 Namespace Prefixes

39 2.3 Terminology

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

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

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

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

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

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

52 3 Usage

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

55 3.1 Processing Model

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

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

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

70 3.2 SAML Version Differences

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

73 3.2.1 Assertion Identifier

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

76 3.2.2 Relationship of Subjects to Statements

77 A SAML assertion contains a collection of 0 or more statements. In SAML V1.1, a separate subject with
78 separate subject confirmation methods may be specified for each statement of an assertion. In SAML

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

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.

79 V2.0, at most one subject and at most one set of subject confirmation methods may be specified for all
80 the statements of the assertion. These distinctions are described in more detail by the following
81 paragraphs.

82 A SAML V1.1 statement that contains a `<saml:Subject>` element (i.e., a subject statement) may
83 contain a `<saml:SubjectConfirmation>` element that defines the rules for confirming the subject and
84 claims of the statement. If present, the `<saml:SubjectConfirmation>` element occurs within the
85 subject element, and defines one or more methods (i.e., `<saml:ConfirmationMethod>` elements) by
86 which the statement may be confirmed and will include a `<ds:KeyInfo>`³ element when any of the
87 specified methods are based on demonstration of a confirmation key. The
88 `<saml:SubjectConfirmation>` element also provides for the inclusion of additional information to be
89 applied in the confirmation method processing via the optional `<saml:SubjectConfirmationData>`
90 element. The following example depicts a SAML V1.1 assertion containing two subject statements with
91 different subjects and different subject confirmation elements.

```
92 <saml:Assertion xmlns:saml="..." xmlns:ds="..."
93     MajorVersion="1" MinorVersion="1" >
94     ...
95     <saml:SubjectStatement>
96     <saml:Subject>
97         <saml:NameIdentifier>
98             ...
99         </saml:NameIdentifier>
100        <saml:SubjectConfirmation>
101            <saml:ConfirmationMethod>
102                urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
103            </saml:ConfirmationMethod>
104            <saml:ConfirmationMethod>
105                urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
106            </saml:ConfirmationMethod>
107            <ds:KeyInfo>
108                <ds:KeyValue>...</ds:KeyValue>
109            </ds:KeyInfo>
110        </saml:SubjectConfirmation>
111    </saml:Subject>
112    ...
113 </saml:SubjectStatement>
114     <saml:SubjectStatement>
115     <saml:Subject>
```

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

```

116     <saml:NameIdentifier>
117         ...
118     </saml:NameIdentifier>
119     <saml:SubjectConfirmation>
120         <saml:ConfirmationMethod>
121             urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
122         </saml:ConfirmationMethod>
123     </saml:SubjectConfirmation>
124 </saml:Subject>
125     ... .
126 </saml:SubjectStatement>
127     ...
128 </saml:Assertion>

```

129 A SAML V2.0 assertion may contain a single `<saml2:Subject>` that applies to all the statements of the
130 assertion. When a subject is included in A SAML V2.0 assertion, it may contain any number of
131 `<saml2:SubjectConfirmation>` elements, satisfying any of which is sufficient to confirm the subject
132 and all the statements of the assertion. Each `<saml2:SubjectConfirmation>` element identifies a
133 single confirmation method (by attribute value) and may include an optional
134 `<saml2:SubjectConfirmationData>` element that is used to specify optional confirmation method
135 independent condition attributes and to define additional method specific confirmation data. In the case of
136 a key dependent confirmation method, a complex schema type,
137 `saml2:KeyInfoConfirmationDataType`, that includes 1 or more `<ds:KeyInfo>` elements, can be
138 specified as the `xsi:type` of the `<saml2:SubjectConfirmationData>` element. In this case,
139 each `<ds:KeyInfo>` element identifies a key that may be demonstrated to confirm the assertion. The
140 following example depicts a SAML V2.0 assertion containing a subject with multiple confirmation
141 elements that apply to all the statements of the assertion.

```

142 <saml2:Assertion xmlns:saml2="..." xmlns:ds="..." xmlns:xsi="...">
143   <saml2:Subject>
144     <saml2:NameID>
145       ...
146     </saml2:NameID>
147     <saml2:SubjectConfirmation
148       Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
149       <saml2:SubjectConfirmationData>
150         Address="129.148.9.42"
151       </saml2:SubjectConfirmationData>
152     </saml2:SubjectConfirmation>
153     <saml2:SubjectConfirmation
154       Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
155       <saml2:SubjectConfirmationData
156         xsi:type="saml2:KeyInfoConfirmationDataType">
157         <ds:KeyInfo>
158           <ds:KeyValue>...</ds:KeyValue>
159         </ds:KeyInfo>
160       </saml2:SubjectConfirmationData>
161     </saml2:SubjectConfirmation>
162   </saml2:Subject>

```

```
163         ...
164         <saml2:Statement>
165         ...
166     </saml2:Statement>
167
168     <saml2:Statement>
169     ...
170 </saml2:Statement>
171     ...
172
173 </saml2:Assertion>
```

174 3.2.3 Assertion URI Reference Replaces AuthorityBinding

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

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

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

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

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

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

184 3.2.4 Attesting Entity Identifier

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

```
187 <saml2:SubjectConfirmation xmlns:saml2="..."
188     Method="urn:oasis:names:tc:SAML:2.0:cm:sender-vouches">
189     <NameID>
190         gateway
191     </NameID>
192     <saml2:SubjectConfirmationData>
193         Address="129.148.9.42"
194     </saml2:SubjectConfirmationData>
195 </saml2:SubjectConfirmation>
```

196 3.3 Attaching Security Tokens

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

```
201 <S12:Envelope xmlns:S12="...">
202     <S12:Header>
203         <wsse:Security xmlns:wsse="...">
204             <saml:Assertion xmlns:saml="...">
```

```

205     AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
206     IssueInstant="2003-04-17T00:46:02Z"
207     Issuer="www.opensaml.org"
208     MajorVersion="1"
209     MinorVersion="1">
210     <saml:AuthenticationStatement>
211         <saml:Subject>
212             <saml:NameIdentifier
213                 NameQualifier="www.example.com"
214                 Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName">
215                 uid=joe,ou=people,ou=saml-demo,o=baltimore.com
216             </saml:NameIdentifier>
217             <saml:SubjectConfirmation>
218                 <saml:ConfirmationMethod>
219                     urn:oasis:names:tc:SAML:1.0:cm:bearer
220                 </saml:ConfirmationMethod>
221             </saml:SubjectConfirmation>
222         </saml:Subject>
223     </saml:AuthenticationStatement>
224
225     </saml:Assertion>
226
227 </wsse:Security>
228 </S12:Header>
229 <S12:Body>
230     . . .
231 </S12:Body>
232 </S12:Envelope>

```

233 3.4 Identifying and Referencing Security Tokens

234 The [WSS: SOAP Message Security](#) specification defines the `<wsse:SecurityTokenReference>`
235 element for referencing security tokens. Three forms of token references are defined by this element and
236 the element schema includes provision for defining additional reference forms should they be necessary.
237 The three forms of token references defined by the `<wsse:SecurityTokenReference>` element are
238 defined as follows:

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

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

248 When a key identifier is used to reference a V1.1 SAML assertion that is not contained in the same
249 message as the key identifier, a `<saml:AuthorityBinding>` element MUST be contained in the
250 `<wsse:SecurityTokenReference>` element containing the key identifier. The contents of the
251 `<saml:AuthorityBinding>` element MUST contain values sufficient for the intended recipients of the
252 `<wsse:SecurityTokenReference>` to acquire the identified assertion from the intended Authority. To
253 this end, the value of the `AuthorityKind` attribute of the `<saml:AuthorityBinding>` element
254 MUST be "samlp:AssertionIdReference".

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

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

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

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

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

272 An Embedded reference – a reference that encapsulates a security token.

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

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

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

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).

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

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

285 A SAML assertion reference may occur as encrypted content within an `<xenc:EncryptedData>`
286 element referenced from a `<xenc:DataReference>` element within an `<xenc:ReferenceList>`
287 element. In this case, the assertion reference (which may contain an embedded assertion) is encrypted.

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

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

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

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

299 Every key identifier, direct, or embedded reference to a SAML assertion SHOULD contain a
300 `wsse11:TokenType` attribute and the value of this attribute MUST be the value from Table 3 that
301 identifies the type and version of the referenced security token. When the referenced assertion is a SAML
302 V2.0 Assertion the reference MUST contain a `wsse11: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

303 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

304 Table-3 TokenType Attribute Values

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

308 3.4.1 SAML Assertion Referenced from Header or Element

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

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

```
316 <S12:Envelope xmlns:S12="...">
317   <S12:Header>
318     <wsse:Security xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="...">
319       <saml:Assertion xmlns:saml="..."
320         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
321         IssueInstant="2003-04-17T00:46:02Z"
322         Issuer="www.opensaml.org"
323         MajorVersion="1"
324         MinorVersion="1">
325     </saml:Assertion>
326     <wsse:SecurityTokenReference wsu:Id="STR1"
327       wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
328 profile-1.1#SAMLV1.1">
329       <wsse:KeyIdentifier wsu:Id="..."
330         ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
331 profile-1.0#SAMLAssertionID">
332         _a75adf55-01d7-40cc-929f-dbd8372ebdfc
333       </wsse:KeyIdentifier>
334     </wsse:SecurityTokenReference>
335   </wsse:Security>
336 </S12:Header>
337 <S12:Body>
338   . . .
339 </S12:Body>
340 </S12:Envelope>
```

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

```
343 <wsse:SecurityTokenReference
344   xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..."
345   wsu:Id="STR1"
346   wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
347 profile-1.1#SAMLV2.0">
348   <wsse:KeyIdentifier wsu:Id="..."
349     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
350 1.1#SAMLID">
351     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
352   </wsse:KeyIdentifier>
353 </wsse:SecurityTokenReference>
```

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

```
358 <wsse:SecurityTokenReference
359   xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..."
```

```

360     wsu:Id="STR1"
361     wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
362 profile-1.1#SAMLV1.1">
363     <saml:AuthorityBinding xmlns:saml="..."
364     Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
365     Location="http://www.opensaml.org/SAML-Authority"
366     AuthorityKind="samlp:AssertionIdReference"/>
367     <wsse:KeyIdentifier
368     wsu:Id="..."
369     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
370 1.0#SAMLAssertionID">
371     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
372     </wsse:KeyIdentifier>
373 </wsse:SecurityTokenReference>

```

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

```

376 <wsse:SecurityTokenReference
377     xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..."
378     wsu:Id="..."
379     wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
380 profile-1.1#SAMLV2.0">
381     <wsse:Reference
382     wsu:Id="..."
383     URI="https://saml.example.edu/assertion-authority?ID=abcde">
384     </wsse:Reference>
385 </wsse:SecurityTokenReference>

```

386 3.4.2 SAML Assertion Referenced from KeyInfo

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

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

```

392 <ds:KeyInfo xmlns:ds="...">
393     <wsse:SecurityTokenReference
394     xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..."
395     wsu:Id="STR1"
396     wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
397 profile-1.1#SAMLV1.1">
398     <wsse:KeyIdentifier wsu:Id="..."
399     ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
400 1.0#SAMLAssertionID">
401     _a75adf55-01d7-40cc-929f-dbd8372ebdfc
402     </wsse:KeyIdentifier>
403     </wsse:SecurityTokenReference>
404 </ds:KeyInfo>

```


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

```
408 <ds:KeyInfo xmlns:ds="...">
409   <wsse:SecurityTokenReference
410     xmlns:wsse="..." xmlns:wsu="..." xmlns:wss11="..."
411     wsu:Id="STR1"
412     wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
413 profile-1.1#SAMLV2.0">
414     <wsse:KeyIdentifier wsu:Id="..."
415       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
416 1.1#SAMLID">
417       _a75adf55-01d7-40cc-929f-dbd8372ebdfc
418     </wsse:KeyIdentifier>
419   </wsse:SecurityTokenReference>
420 </ds:KeyInfo>
```

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

```
425 <ds:KeyInfo xmlns:ds="...">
426   <wsse:SecurityTokenReference
427     xmlns:wsse="..." xmlns:wsu="..." xmlns:wss11="..."
428     wsu:Id="STR1"
429     wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
430 profile-1.1#SAMLV1.1">
431     <saml:AuthorityBinding xmlns:saml="..."
432       Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
433       Location="http://www.opensaml.org/SAML-Authority"
434       AuthorityKind="samlp:AssertionIdReference"/>
435     <wsse:KeyIdentifier wsu:Id="..."
436       ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
437 1.0#SAMLAssertionID">
438       _a75adf55-01d7-40cc-929f-dbd8372ebdfc
439     </wsse:KeyIdentifier>
440   </wsse:SecurityTokenReference>
441 </ds:KeyInfo>
```

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

```
444 <ds:KeyInfo xmlns:ds="...">
445   <wsse:SecurityTokenReference
446     xmlns:wsse="..." xmlns:wsu="..." xmlns:wss11="..."
447     wsu:id="STR1"
448     wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
449 profile-1.1#SAMLV2.0">
450     <wsse:Reference
451       wsu:id="..."
```

452 URI="https://saml.example.edu/assertion-authority?ID=abcde">
453 </wsse:Reference>
454 </wsse:SecurityTokenReference>
455 </ds:KeyInfo>
456 <ds:KeyInfo> elements may also occur in <xenc:EncryptedData> and <xenc:EncryptedKey>
457 elements where they serve to identify the encryption key. <ds:KeyInfo> elements may also occur in
458 SAML SubjectConfirmation elements where they identify a key that MUST be demonstrated to
459 confirm the subject of the corresponding statement(s).
460 Conformant implementations of this profile are NOT required to process SAML assertion references
461 occurring within the <ds:KeyInfo> elements within <xenc:EncryptedData>,
462 <xenc:EncryptedKey>, or SAML SubjectConfirmation elements.

463 3.4.3 SAML Assertion Referenced from SignedInfo

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

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

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

```
480 <wsse:SecurityTokenReference
481     xmlns:wsse="..." xmlns:wsu="..." xmlns:wssell="..." wsu:Id="STR1"
482     wssell:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
483     profile-1.1#SAMLV1.1">
484     <saml:AuthorityBinding xmlns:saml="..."
485         Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
486         Location="http://www.opensaml.org/SAML-Authority"
487         AuthorityKind="samlp:AssertionIdReference"/>
488     <wsse:KeyIdentifier wsu:Id="..."
489         ValueType="http://docs.oasis-open.org/wss/oasis-2004XX-wss-saml-token-
490     profile-1.0#SAMLAssertionID">
491         _a75adf55-01d7-40cc-929f-dbd8372ebdfc
492     </wsse:KeyIdentifier>
493 </wsse:SecurityTokenReference>
494
495 <ds:SignedInfo xmlns:ds="..." xmlns:wsse="...">
496     <ds:CanonicalizationMethod
497         Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
498     <ds:SignatureMethod
```

```

499     Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
500   <ds:Reference URI="#STR1">
501     <Transforms>
502       <ds:Transform
503         Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-
504 message-security-1.0#STR-Transform">
505         <wsse:TransformationParameters>
506           <ds:CanonicalizationMethod
507             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
508           </wsse:TransformationParameters>
509         </ds:Transform>
510       </Transforms>
511     <ds:DigestMethod
512       Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
513
514   <ds:DigestValue>...</ds:DigestValue>
515 </ds:Reference>
516 </ds:SignedInfo>

```

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

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

525 3.4.4 SAML Assertion Referenced from Encrypted Data Reference

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

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

```

538 <xenc:EncryptedData xmlns:xenc="..." xmlns:ds="..." Id="EncryptedSTR1">
539   <ds:KeyInfo>
540     . . .
541   </ds:KeyInfo>
542   <xenc:CipherData>
543     <xenc:CipherValue>...</xenc:CipherValue>
544   </xenc:CipherData>
545 </xenc:EncryptedData>

```

546 <xenc:ReferenceList xmlns:xenc="...">
 547 <xenc:DataReference URI="#EncryptedSTR1"/>
 548 </xenc:ReferenceList>

549 3.4.5 SAML Version Support and Backward Compatibility

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

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

559 3.5 Subject Confirmation of SAML Assertions

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

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

570 The following table enumerates the mandatory subject confirmation methods and summarizes their
 571 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.

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

577 **3.5.1 Holder-of-key Subject Confirmation Method**

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

581 **3.5.1.1 Attesting Entity**

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

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

589 To satisfy the associated confirmation method processing to be performed by the message receiver, the
590 attesting entity MUST demonstrate knowledge of the confirmation key. The attesting entity MAY
591 accomplish this by using the confirmation key to sign content within the message and by including the
592 resulting `<ds:Signature>` element in the `<wsse:Security>` header. `<ds:Signature>` elements
593 produced for this purpose MUST conform to the `canonicalization` and token pre-pending rules
594 defined in the [WSS: SOAP Message Security](#) specification. The attesting entity MAY protect against
595 substitution of a different but equivalently confirmed⁶ assertion by including, as described in section 3.4.3
596 "SAML Assertion Referenced from SignedInfo", the SAML assertion (or an unambiguous reference to it)
597 in the content signed to demonstrate knowledge of the confirmation key.

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

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

605 **3.5.1.2 Receiver**

606 Of the SAML assertions it selects for processing, a message receiver MUST NOT accept statements of
607 these assertions based on a holder-of-key `SubjectConfirmation` element defined for the statements

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)`.

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

608 (within the assertion) unless the receiver has validated the integrity of the assertion and the attesting
609 entity has demonstrated knowledge of a key identified within the confirmation element.

610 If the receiver determines that the attesting entity has demonstrated knowledge of a subject confirmation
611 key, then the subjects and claims of the SAML statements confirmed by the key MAY be attributed to the
612 attesting entity and any content of the message (including any SAML statements) whose integrity is
613 protected by the key MAY be considered to have been provided by the attesting entity.

614 3.5.1.3 Example V1.1

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

```
618 <?xml version="1.0" encoding="UTF-8"?>
619 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
620   <S12:Header>
621     <wsse:Security xmlns:wsse="..." xmlns:wssell="..." xmlns:ds="...">
622       <saml:Assertion xmlns:saml="..."
623         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
624         IssueInstant="2005-05-27T16:53:33.173Z"
625         Issuer="www.opensaml.org"
626         MajorVersion="1"
627         MinorVersion="1">
628         <saml:Conditions
629           NotBefore="2005-05-27T16:53:33.173Z"
630           NotOnOrAfter="2005-05-27T16:58:33.17302Z"/>
631         <saml:AttributeStatement>
632           <saml:Subject>
633             <saml:NameIdentifier
634               NameQualifier="www.example.com"
635               Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName">
636               uid=joe,ou=people,ou=saml-demo,o=baltimore.com
637             </saml:NameIdentifier>
638             <saml:SubjectConfirmation>
639               <saml:ConfirmationMethod>
640                 urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
641               </saml:ConfirmationMethod>
642               <ds:KeyInfo>
643                 <ds:KeyValue>...</ds:KeyValue>
644               </ds:KeyInfo>
645             </saml:SubjectConfirmation>
646           </saml:Subject>
647           <saml:Attribute
648             AttributeName="MemberLevel"
649             AttributeNamespace="http://www.oasis-open.org/Catalyst2002/attributes">
650             <saml:AttributeValue>gold</saml:AttributeValue>
651           </saml:Attribute>
652         </saml:AttributeStatement>
653       </saml:Assertion>
654     </wsse:Security>
655   </S12:Header>
656 </S12:Envelope>
```

```

654         AttributeName="E-mail"
655         AttributeNamespace="http://www.oasis-open.org/Catalyst2002/attributes">
656         <saml:AttributeValue>joe@yahoo.com</saml:AttributeValue>
657     </saml:Attribute>
658 </saml:AttributeStatement>
659 <ds:Signature>...</ds:Signature>
660 </saml:Assertion>
661
662 <ds:Signature>
663     <ds:SignedInfo>
664         <ds:CanonicalizationMethod
665             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
666         <ds:SignatureMethod
667             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
668         <ds:Reference
669             URI="#MsgBody">
670             <ds:DigestMethod
671                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
672             <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
673         </ds:Reference>
674     </ds:SignedInfo>
675     <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
676     <ds:KeyInfo>
677         <wsse:SecurityTokenReference wsu:Id="STR1"
678             wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
679 token-profile-1.1#SAMLV1.1">
680             <wsse:KeyIdentifier wsu:Id="..."
681                 ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
682 profile-1.0#SAMLAssertionID">
683                 _a75adf55-01d7-40cc-929f-dbd8372ebdfc
684             </wsse:KeyIdentifier>
685         </wsse:SecurityTokenReference>
686     </ds:KeyInfo>
687 </ds:Signature>
688 </wsse:Security>
689 </S12:Header>
690
691 <S12:Body wsu:Id="MsgBody">
692     <ReportRequest>
693         <TickerSymbol>SUNW</TickerSymbol>
694     </ReportRequest>
695 </S12:Body>
696 </S12:Envelope>

```

697 3.5.1.4 Example V2.0

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

700 <wsse:Security> header:

```
701 <?xml version="1.0" encoding="UTF-8"?>
702 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
703   <S12:Header>
704
705     <wsse:Security xmlns:wsse="..." xmlns:wssell="..." xmlns:ds="...">
706       <saml2:Assertion xmlns:saml2="..." xmlns:xsi="..."
707         ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
708         <saml2:Subject>
709           <saml2:NameID>
710             ...
711           </saml2:NameID>
712           <saml2:SubjectConfirmation
713             Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
714             <saml2:SubjectConfirmationData
715               xsi:type="saml2:KeyInfoConfirmationDataType">
716               <ds:KeyInfo>
717                 <ds:KeyValue>...</ds:KeyValue>
718               </ds:KeyInfo>
719             </saml2:SubjectConfirmationData>
720           </saml2:SubjectConfirmation>
721         </saml2:Subject>
722         <saml2:Statement>
723           ...
724         </saml2:Statement>
725         <ds:Signature>...</ds:Signature>
726       </saml2:Assertion>
727
728     <ds:Signature>
729       <ds:SignedInfo>
730         <ds:CanonicalizationMethod
731           Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
732         <ds:SignatureMethod
733           Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
734         <ds:Reference
735           URI="#MsgBody">
736           <ds:DigestMethod
737             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
738           <ds:DigestValue>GyGsF0Pi4xPU...</ds:DigestValue>
739         </ds:Reference>
740       </ds:SignedInfo>
741       <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
742     </ds:Signature>
743   </S12:Header>
744 </S12:Envelope>
```



```

743         <wsse:SecurityTokenReference wsu:Id="STR1"
744             wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
745 token-profile-1.1#SAMLV2.0">
746             <wsse:KeyIdentifier wsu:Id="..."
747                 ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
748 profile-1.1#SAMLID">
749                 _a75adf55-01d7-40cc-929f-dbd8372ebdfc
750             </wsse:KeyIdentifier>
751         </wsse:SecurityTokenReference>
752     </ds:KeyInfo>
753 </ds:Signature>
754 </wsse:Security>
755 </S12:Header>
756
757 <S12:Body wsu:Id="MsgBody">
758     <ReportRequest>
759         <TickerSymbol>SUNW</TickerSymbol>
760     </ReportRequest>
761 </S12:Body>
762 </S12:Envelope>

```

763 3.5.2 Sender-vouches Subject Confirmation Method

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

767 3.5.2.1 Attesting Entity

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

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

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

782 3.5.2.2 Receiver

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

788 3.5.2.3 Example V1.1

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

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

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

```
800 <?xml version="1.0" encoding="UTF-8"?>
801 <S12:Envelope xmlns:S12="..." xmlns:wssu="...">
802
803   <S12:Header>
804     <wsse:Security xmlns:wsse="..." xmlns:wssell="..." xmlns:ds="...">
805
806       <saml:Assertion xmlns:saml="..."
807         AssertionID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc"
808         IssueInstant="2005-05-27T16:53:33.173Z"
809         Issuer="www.opensaml.org"
810         MajorVersion="1"
811         MinorVersion="1">
812         <saml:Conditions
813           NotBefore="2005-05-27T16:53:33.173Z"
814           NotOnOrAfter="2005-05-27T16:58:33.173Z"/>
815         <saml:AttributeStatement>
816           <saml:Subject>
817             <saml:NameIdentifier
818               NameQualifier="www.example.com"
819               Format="urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName">
820               uid=proxy,ou=system,ou=saml-demo,o=baltimore.com
821             </saml:NameIdentifier>
822             <saml:SubjectConfirmation>
823               <saml:ConfirmationMethod>
824                 urn:oasis:names:tc:SAML:1.0:cm:holder-of-key
825               </saml:ConfirmationMethod>
826               <ds:KeyInfo>
827                 <ds:KeyValue>...</ds:KeyValue>
828               </ds:KeyInfo>
829             </saml:SubjectConfirmation>
830           </saml:Subject>
831           <saml:Attribute>
832             . . .
833           </saml:Attribute>
834           . . .
```

```

835     </saml:AttributeStatement>
836 </saml:Assertion>
837
838     <wsse:SecurityTokenReference wsu:Id="STR1">
839         wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
840 profile-1.1#SAMLV1.1">
841             <saml:AuthorityBinding xmlns:saml="..."
842                 Binding="urn:oasis:names:tc:SAML:1.0:bindings:SOAP-binding"
843                 Location="http://www.opensaml.org/SAML-Authority"
844                 AuthorityKind="samlp:AssertionIdReference"/>
845             <wsse:KeyIdentifier wsu:Id="..."
846                 ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
847 profile-1.0#SAMLAssertionID">
848                 _a75adf55-01d7-40cc-929f-dbd8372ebdbe
849             </wsse:KeyIdentifier>
850 </wsse:SecurityTokenReference>
851
852 <ds:Signature>
853     <ds:SignedInfo>
854         <ds:CanonicalizationMethod
855             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
856         <ds:SignatureMethod
857             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
858     <ds:Reference URI="#STR1">
859         <Transforms>
860             <ds:Transform
861                 Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
862 soap-message-security-1.0#STR-Transform">
863                 <wsse:TransformationParameters>
864                     <ds:CanonicalizationMethod
865                         Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
866                     </wsse:TransformationParameters>
867                 </ds:Transform>
868             </Transforms>
869             <ds:DigestMethod
870                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
871             <ds:DigestValue>...</ds:DigestValue>
872         </ds:Reference>
873     <ds:Reference URI="#MsgBody">
874         <ds:DigestMethod
875             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
876         <ds:DigestValue>...</ds:DigestValue>
877     </ds:Reference>
878 </ds:SignedInfo>
879 <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
880 <ds:KeyInfo>
881     <wsse:SecurityTokenReference wsu:Id="STR2">

```

```

882         wss11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
883 token-profile-1.1#SAMLV1.1">
884         <wsse:KeyIdentifier wsu:Id="..."
885         Value="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
886 profile-1.0#SAMLAssertionID">
887             _a75adf55-01d7-40cc-929f-dbd8372ebdfc
888         </wsse:KeyIdentifier>
889     </wsse:SecurityTokenReference>
890 </ds:KeyInfo>
891 </ds:Signature>
892 </wsse:Security>
893 </S12:Header>
894
895 <S12:Body wsu:Id="MsgBody">
896     <ReportRequest>
897         <TickerSymbol>SUNW</TickerSymbol>
898     </ReportRequest>
899 </S12:Body>
900 </S12:Envelope>

```

901 3.5.2.4 Example V2.0

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

```

903 <?xml version="1.0" encoding="UTF-8"?>
904 <S12:Envelope xmlns:S12="..." xmlns:wsu="...">
905     <S12:Header>
906
907         <wsse:Security xmlns:wsse="..." xmlns:wss11="..." xmlns:ds="...">
908             <saml2:Assertion xmlns:saml2="..." xmlns:xsi="..."
909
910                 ID="_a75adf55-01d7-40cc-929f-dbd8372ebdfc">
911                 <saml2:Subject>
912                     <saml2:NameID>
913                         ...
914                     </saml2:NameID>
915                     <saml2:SubjectConfirmation
916                         Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
917                         <saml2:SubjectConfirmationData
918                             xsi:type="saml2:KeyInfoConfirmationDataType">
919                             <ds:KeyInfo>
920                                 <ds:KeyValue>...</ds:KeyValue>
921                             </ds:KeyInfo>
922                         </saml2:SubjectConfirmationData>
923                     </saml2:SubjectConfirmation>
924                 </saml2:Subject>
925                 <saml2:Statement>
926                     ...

```

```

927         </saml2:Statement>
928         <ds:Signature>...</ds:Signature>
929     </saml2:Assertion>
930
931     <wsse:SecurityTokenReference wsu:Id="STR1"
932         wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
933 profile-1.1#SAMLV2.0">
934         <wsse:Reference wsu:Id="..."
935             URI="https://www.opensaml.org?_a75adf55-01d7-40cc-929f-dbd8372ebdbe">
936             </wsse:Reference>
937         </wsse:SecurityTokenReference>
938
939     <ds:Signature>
940     <ds:SignedInfo>
941         <ds:CanonicalizationMethod
942             Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
943         <ds:SignatureMethod
944             Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
945     <ds:Reference URI="#STR1">
946         <Transforms>
947             <ds:Transform
948
949                 Algorithm="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-
950 message-security-1.0#STR-Transform">
951                 <wsse:TransformationParameters>
952                     <ds:CanonicalizationMethod
953                         Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
954                     </wsse:TransformationParameters>
955                 </ds:Transform>
956             </Transforms>
957             <ds:DigestMethod
958                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
959             <ds:DigestValue>...</ds:DigestValue>
960         </ds:Reference>
961     <ds:Reference URI="#MsgBody">
962         <ds:DigestMethod
963             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
964         <ds:DigestValue>...</ds:DigestValue>
965     </ds:Reference>
966 </ds:SignedInfo>
967 <ds:SignatureValue>HJJWbvqW9E84vJVQk...</ds:SignatureValue>
968 <ds:KeyInfo>
969     <wsse:SecurityTokenReference wsu:Id="STR2"
970         wsse11:TokenType="http://docs.oasis-open.org/wss/oasis-wss-saml-
971 token-profile-1.1#SAMLV2.0">
972     <wsse:KeyIdentifier wsu:Id="..."

```

```

973         ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
974 profile-1.1#SAMLID">
975         _a75adf55-01d7-40cc-929f-dbd8372ebdfc
976         </wsse:KeyIdentifier>
977     </wsse:SecurityTokenReference>
978 </ds:KeyInfo>
979 </ds:Signature>
980 </wsse:Security>
981 </S12:Header>
982
983 <S12:Body wsu:Id="MsgBody">
984     <ReportRequest>
985         <TickerSymbol>SUNW</TickerSymbol>
986     </ReportRequest>
987 </S12:Body>
988 </S12:Envelope>

```

989 3.5.3 Bearer Confirmation Method

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

996 3.6 Error Codes

997 When a system that implements the SAML token profile of [WSS: SOAP Message Security](#) does not
998 perform its normal processing because of an error detected during the processing of a security header, it
999 MAY choose to report the cause of the error using the SOAP fault mechanism. The SAML token profile of
1000 [WSS: SOAP Message Security](#) does not require that SOAP faults be returned for such errors, and
1001 systems that choose to return faults SHOULD take care not to introduce any security vulnerabilities as a
1002 result of the information returned in error responses.

1003 Systems that choose to return faults SHOULD respond with the error codes and fault strings defined in
1004 the [WSS: SOAP Message Security](#) specification. The RECOMMENDED correspondence between the
1005 common assertion processing failures and the error codes defined in [WSS: SOAP Message Security](#) are
1006 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

1007 The preceding table defines fault codes in a form suitable for use with SOAP 1.1. The [WSS: SOAP](#)
1008 [Message Security](#) specification describes how to map SOAP 1.1 fault constructs to the SOAP 1.2 fault
1009 constructs.

1010 4 Threat Model and Countermeasures (non- 1011 normative)

1012 This document defines the mechanisms and procedures for securely attaching SAML assertions to SOAP
1013 messages. SOAP messages are used in multiple contexts, specifically including cases where the
1014 message is transported without an active session, the message is persisted, or the message is routed
1015 through a number of intermediaries. Such a general context of use suggests that users of this profile must
1016 be concerned with a variety of threats.

1017 In general, the use of SAML assertions with [WSS: SOAP Message Security](#) introduces no new threats
1018 beyond those identified for SAML or by the [WSS: SOAP Message Security](#) specification. The following
1019 sections provide an overview of the characteristics of the threat model, and the countermeasures that
1020 SHOULD be adopted for each perceived threat.

1021 4.1 Eavesdropping

1022 Eavesdropping is a threat to the SAML token profile of [WSS: SOAP Message Security](#) in the same
1023 manner as it is a threat to any network protocol. The routing of SOAP messages through intermediaries
1024 increases the potential incidences of eavesdropping. Additional opportunities for eavesdropping exist
1025 when SOAP messages are persisted.

1026 To provide maximum protection from eavesdropping, assertions, assertion references, and sensitive
1027 message content SHOULD be encrypted such that only the intended audiences can view their content.
1028 This approach removes threats of eavesdropping in transit, but MAY not remove risks associated with
1029 storage or poor handling by the receiver.

1030 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1031 references from eavesdropping while in transport, but message content MUST be encrypted above the
1032 transport if it is to be protected from eavesdropping by intermediaries.

1033 4.2 Replay

1034 Reliance on authority-protected (e.g., signed) assertions with a holder-of-key subject confirmation
1035 mechanism precludes all but a holder of the key from binding the assertions to a SOAP message.
1036 Although this mechanism effectively restricts data origin to a holder of the confirmation key, it does not, by
1037 itself, provide the means to detect the capture and resubmission of the message by other parties.

1038 Assertions that contain a sender-vouches confirmation mechanism introduce another dimension to replay
1039 vulnerability if the assertions impose no restriction on the entities that may use or reuse the assertions.

1040 Replay attacks can be detected by receivers if message senders include additional message identifying
1041 information (e.g., timestamps, nonces, and or recipient identifiers) within origin-protected message
1042 content and receivers check this information against previously received values.

1043 4.3 Message Insertion

1044 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message insertion attacks.

1045 4.4 Message Deletion

1046 The SAML token profile of [WSS: SOAP Message Security](#) is not vulnerable to message deletion attacks.

1047 4.5 Message Modification

1048 Messages constructed according to this specification are protected from message modification if
1049 receivers can detect unauthorized modification of relevant message content. Therefore, it is strongly
1050 RECOMMENDED that all relevant and immutable message content be signed by an attesting entity.

1051 Receivers SHOULD only consider the correspondence between the subject of the SAML assertions and
1052 the SOAP message content to have been established for those portions of the message that are
1053 protected by the attesting entity against modification by another entity.

1054 To ensure that message receivers can have confidence that received assertions have not been forged or
1055 altered since their issuance, SAML assertions appearing in or referenced from `<wsse:Security>`
1056 header elements MUST be protected against unauthorized modification (e.g., signed) by their issuing
1057 authority or the attesting entity (as the case warrants). It is strongly RECOMMENDED that an attesting
1058 entity sign any `<saml:Assertion>` elements that it is attesting for and that are not signed by their
1059 issuing authority.

1060 Transport-layer security MAY be used to protect the message and contained SAML assertions and/or
1061 assertion references from modification while in transport, but signatures are required to extend such
1062 protection through intermediaries.

1063 To ensure that message receivers can have confidence that an assertion with an equivalent confirmation
1064 key has not been substituted for the assertion used by the attesting entity, the attesting entity MAY
1065 include the assertion (or an unambiguous reference to it) in the attested for (i.e., signed) message
1066 content.

1067 **4.6 Man-in-the-Middle**

1068 Assertions with a holder-of-key subject confirmation method are not vulnerable to a MITM attack.
1069 Assertions with a sender-vouches subject confirmation method are vulnerable to MITM attacks to the
1070 degree that the receiver does not have a trusted binding of key to the attesting entity's identity.

5 References

- 1071
- 1072 **[GLOSSARY]** Informational RFC 2828, "[Internet Security Glossary](#)," May 2000.
- 1073 **[KEYWORDS]** S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," [RFC 2119](#), Harvard University, March 1997
- 1074
- 1075 **[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.
- 1076
- 1077
- 1078 **[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.
- 1079
- 1080
- 1081 **[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.
- 1082
- 1083
- 1084 **[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.
- 1085
- 1086
- 1087 **[SOAP]** W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.
- 1088 W3C Working Draft, Nilo Mitra (Editor), [SOAP Version 1.2 Part 0: Primer](#), June 2002.
- 1089
- 1090 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-
- 1091 Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 1: Messaging Framework](#), June 2002.
- 1092
- 1093 W3C Working Draft, Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-
- 1094 Jacques Moreau, Henrik Frystyk Nielsen (Editors), [SOAP Version 1.2 Part 2: Adjuncts](#), June 2002.
- 1095
- 1096 **[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.
- 1097
- 1098
- 1099 **[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.
- 1100
- 1101 **[WSS: SAML Token Profile]** Oasis Standard, P. Hallam-Baker, A. Nadalin, C. Kaler, R. Monzillo (Editors), [Web Services Security: SAML Token Profile 1.0](#), December 2004.
- 1102
- 1103 **[WSS: SOAP Message Security V1.0]** Oasis Standard, A. Nadalin, C.Kaler, P. Hallam-Baker, R. Monzillo (Editors), [Web Services Security: SOAP Message Security 1.0 \(WS-Security 2004\)](#), August 2003.
- 1104
- 1105
- 1106 **[WSS: SOAP Message Security V1.1]** Oasis Standard, A. Nadalin, C.Kaler, R. Monzillo, P. Hallam-Baker,(Editors), [Web Services Security: SOAP Message Security 1.1 \(WS-Security 2004\)](#), December 2005.
- 1107
- 1108
- 1109 **[WSS: SOAP Message Security V1.1.1]** [Web Services Security: SOAP Message Security Version 1.1.1](#). 30 September 2011. OASIS Committee Specification 01. <http://docs.oasis-open.org/wss-m/wss/v1.1.1/cs01/wss-SOAPMessageSecurity-v1.1.1-cs01.html>.
- 1110
- 1111
- 1112 **[XML-ns]** W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.
- 1113 **[XML Signature]** W3C Recommendation, "[XML Signature Syntax and Processing](#)," 12 February 2002.
- 1114
- 1115 **[XML Token]** Contribution to the WSS TC, Chris Kaler (Editor),
- 1116 WS-Security Profile for XML-based Tokens, August 2002.

1117 **6 Conformance**

1118 An implementation conforms to this specification if it meets the requirements in Sections 2.1, 2.2 and 3.

1119 **A. Acknowledgements**

1120 The following individuals have participated in the creation of this specification and are gratefully
 1121 acknowledged:

1122 **Participants:**
 1123 **Current Contributors:**

Tom	Rutt	Fujitsu Limited
Jacques	Durand	Fujitsu Limited
Calvin	Powers	IBM
Kelvin	Lawrence	IBM
Michael	McIntosh	Individual
Thomas	Hardjono	M.I.T.
David	Turner	Microsoft Corporation
Anthony	Nadalin	Microsoft Corporation
Monica	Martin	Microsoft Corporation
Marc	Goodner	Microsoft Corporation
Peter	Davis	Neustar
Hal	Lockhart	Oracle Corporation
Rich	Levinson	Oracle Corporation
Anil	Saldhana	Red Hat
Martin	Raepple	SAP AG
Federico	Rossini	Telecom Italia S.p.a.
Carlo	Milono	TIBCO Software Inc.
Don	Adams	TIBCO Software Inc.
Jerry	Smith	US Department of Defense (DoD)

1124 **Previous Contributors:**

Michael	Hu	Actional
Maneesh	Sahu	Actional
Duane	Nickull	Adobe Systems
Gene	Thurston	AmberPoint
Frank	Siebenlist	Argonne National Laboratory
Peter	Dapkus	BEA
Hal	Lockhart	BEA Systems
Denis	Pilipchuk	BEA Systems
Corinna	Witt	BEA Systems
Steve	Anderson	BMC Software
Rich	Levinson	Computer Associates
Thomas	DeMartini	ContentGuard
Guillermo	Lao	ContentGuard
TJ	Pannu	ContentGuard

Xin	Wang	ContentGuard
Merlin	Hughes	Cybertrust
Dale	Moberg	Cyclone Commerce
Shawn	Sharp	Cyclone Commerce
Rich	Salz	Datapower
Ganesh	Vaideeswaran	Documentum
Sam	Wei	EMC
Tim	Moses	Entrust
Carolina	Canales-Valenzuela	Ericsson
Dana S.	Kaufman	Forum Systems
Toshihiro	Nishimura	Fujitsu
Tom	Rutt	Fujitsu
Kefeng	Chen	GeoTrust
Irving	Reid	Hewlett-Packard
Kojiro	Nakayama	Hitachi
Yutaka	Kudo	Hitachi
Jason	Rouault	HP
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
Bob	Blakley	IBM
Joel	Farrell	IBM
Satoshi	Hada	IBM
Hiroshi	Maruyama	IBM
David	Melgar	IBM
Kent	Tamura	IBM
Wayne	Vicknair	IBM
Don	Flinn	Individual
Phil	Griffin	Individual
Mark	Hayes	Individual
John	Hughes	Individual
Peter	Rostin	Individual
Davanum	Srinivas	Individual
Bob	Morgan	Individual/Internet2
Kate	Cherry	Lockheed Martin
Paul	Cotton	Microsoft
Vijay	Gajjala	Microsoft
Martin	Gudgin	Microsoft
Chris	Kaler	Microsoft
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
Frederick	Hirsch	Nokia
Senthil	Sengodan	Nokia
Abbie	Barbir	Nortel
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Prateek	Mishra	Oracle
Vamsi	Motukuru	Oracle
Ramana	Turlapi	Oracle
Vipin	Samar	Oracle
Jerry	Schwarz	Oracle
Eric	Gravengaard	Reactivity
Andrew	Nash	Reactivity
Stuart	King	Reed Elsevier
Ben	Hammond	RSA Security
Rob	Philpott	RSA Security
Martijn	de Boer	SAP
Blake	Dournaee	Sarvega
Sundeep	Peechu	Sarvega
Coumara	Radja	Sarvega
Pete	Wenzel	SeeBeyond
Jonathan	Tourzan	Sony
Yassir	Elley	Sun
Manveen	Kaur	Sun Microsystems
Ronald	Monzillo	Sun Microsystems
Jan	Alexander	Systinet
Michael	Nguyen	The IDA of Singapore
Don	Adams	TIBCO
Symon	Chang	TIBCO Software
John	Weiland	US Navy
Hans	Granqvist	VeriSign
Phillip	Hallam-Baker	VeriSign
Hemma	Prafullchandra	VeriSign
Morten	Jorgensen	Vordel

1125

1126

B. Revision History

1127

Revision	Date	Editor	Changes Made
WD01	17-January-2011	Carlo Milono	Corrected/added hyperlinks where missing; added Status section
WD02	8-February-2011	Carlo Milono	Added Related Work to reflect v1.1.1 of the specs; changed References for SOAP Message Security to reflect v1.1.1; Changed WD# to 2; Added Date; Moved Current Members to Previous and added new Current Members; saved document under wd02; entered the Revision History Merged Old Current Contributors with Old Previous, created a New Current Contributors.
WD03	16-March-2011	David Turner	Corrected some links
CSD01	2-May-2011	TC Admin	Generated from WD03
CSD02-draft	16-May-11	David Turner	Added conformance statement and corrected a few formatting issues.

1128