



WS-SecurityPolicy v1.2

Editors Draft 01, 19 June 2006

Artifact Identifier:

ws-securitypolicy-1.2-spec-ed-01

Location:

Current: docs.oasis-open.org/ws-sx/200512/ws-securitypolicy

This Version: docs.oasis-open.org/ws-sx/200512/ws-securitypolicy

Previous Version: n/a

Artifact Type:

specification

Technical Committee:

OASIS Web Services Secure Exchange TC

Chair(s):

Kelvin Lawrence, IBM

Chris Kaler, Microsoft

Editor(s):

Anthony Nadalin, IBM

Martin Gudgin, Microsoft

Abbie Barbir, Nortel

Hans Granqvist, VeriSign

OASIS Conceptual Model topic area:

[Topic Area]

Related work:

N/A

Abstract:

This document indicates the policy assertions for use with [WS-Policy](#) which apply to WSS: SOAP Message Security, [WS-Trust](#) and [WS-SecureConversation](#)

Status:

This document was last revised or approved by the WS-SX TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

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

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (www.oasis-open.org/committees/ws-sx/ipr.php).

The non-normative errata page for this specification is located at www.oasis-open.org/committees/ws-sx.

Notices

Copyright © OASIS Open 2006. 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.

Table of Contents

1	Introduction.....	6
1.1	Example.....	6
1.2	Namespaces	7
1.3	Schema Files	8
1.4	Terminology	8
1.4.1	Notational Conventions	8
1.5	Normative References	9
1.6	Non-Normative References	9
2	Security Policy Model	11
2.1	Security Assertion Model	11
2.2	Nested Policy Assertions.....	12
2.3	Security Binding Abstraction	12
3	Policy Considerations	14
3.1	Nested Policy	14
3.1.1	Nesting Policy Elements	14
3.1.2	Nested Policy Assertions.....	15
3.1.3	Nesting Policy Processing Rules.....	16
3.1.4	Nested Policy Normalization Worked Example	16
3.1.5	Nested Policy Intersection Worked Example.....	17
3.2	Policy Subjects.....	19
4	Protection Assertions	21
4.1	Integrity Assertions	21
4.1.1	SignedParts Assertion	21
4.1.2	SignedElements Assertion	22
4.2	Confidentiality Assertions	23
4.2.1	EncryptedParts Assertion.....	23
4.2.2	EncryptedElements Assertion	24
4.3	Required Elements Assertion.....	24
4.3.1	RequiredElements Assertion	25
5	Token Assertions	26
5.1	Token Inclusion.....	26
5.1.1	Token Inclusion Values	26
5.1.2	Token Inclusion and Token References.....	27
5.2	Token Properties	27
5.2.1	[Derived Keys] Property	27
5.2.2	[Explicit Derived Keys] Property	27
5.2.3	[Implicit Derived Keys] Property	27
5.3	Token Assertion Types	28
5.3.1	UsernameToken Assertion	28
5.3.2	IssuedToken Assertion	29
5.3.3	X509Token Assertion	30

5.3.4	KerberosToken Assertion	32
5.3.5	SpnegoContextToken Assertion	33
5.3.6	SecurityContextToken Assertion	34
5.3.7	SecureConversationToken Assertion	36
5.3.8	SamlToken Assertion	38
5.3.9	RelToken Assertion	39
5.3.10	HttpsToken Assertion	41
6	Security Binding Properties	42
6.1	[Algorithm Suite] Property	42
6.2	[Timestamp] Property	44
6.3	[Protection Order] Property	44
6.4	[Signature Protection] Property	44
6.5	[Token Protection] Property	45
6.6	[Entire Header and Body Signatures] Property	45
6.7	[Security Header Layout] Property	45
6.7.1	Strict Layout Rules	46
7	Security Binding Assertions	47
7.1	AlgorithmSuite Assertion	47
7.2	Layout Assertion	49
7.3	TransportBinding Assertion	50
7.4	SymmetricBinding Assertion	51
7.5	AsymmetricBinding Assertion	52
8	Supporting Tokens	56
8.1	SupportingTokens Assertion	57
8.2	SignedSupportingTokens Assertion	58
8.3	EndorsingSupportingTokens Assertion	60
8.4	SignedEndorsingSupportingTokens Assertion	62
8.5	Interaction between [Token Protection] property and supporting token assertions	64
8.6	Example	64
9	WSS: SOAP Message Security Options	66
9.1	Wss10 Assertion	67
9.2	Wss11 Assertion	68
10	WS-Trust Options	70
10.1	Trust10 Assertion	71
11	Guidance on creating new assertions and assertion extensibility	72
11.1	General Design Points	72
11.2	Detailed Design Guidance	72
12	Security Considerations	74
A.	Assertions and WS-PolicyAttachment	75
A.1	Endpoint Policy Subject Assertions	75
A.1.1	Security Binding Assertions	75
A.1.2	Token Assertions	75
A.1.3	WSS: SOAP Message Security 1.0 Assertions	75

A.1.4 WSS: SOAP Message Security 1.1 Assertions	75
A.1.5 Trust 1.0 Assertions	75
A.2 Operation Policy Subject Assertions	75
A.2.1 Supporting Token Assertions	75
A.3 Message Policy Subject Assertions	76
A.3.1 Supporting Token Assertions	76
A.3.2 Protection Assertions	76
A.4 Assertions With Undefined Policy Subject	76
A.4.1 General Assertions	76
A.4.2 Token Usage Assertions	76
A.4.3 Token Assertions.....	77
A.4.4 WSS: SOAP Message Security 1.0 Assertions	77
A.4.5 WSS: SOAP Message Security 1.1 Assertions	77
A.4.6 Trust 1.0 Assertions	77
B. Issued Token Policy	78
C. Strict Security Header Layout Examples	80
C.1 Transport Binding.....	80
C.1.1 Policy	80
C.1.2 Initiator to Recipient Messages	81
C.1.3 Recipient to Initiator Messages	83
C.2 Symmetric Binding	85
C.2.1 Policy	85
C.2.2 Initiator to Recipient Messages	87
C.2.3 Recipient to Initiator Messages	90
C.3 Asymmetric Binding.....	94
C.3.1 Policy	94
C.3.2 Initiator to Recipient Messages	96
C.3.3 Recipient to Initiator Messages	99
D. Acknowledgements	104
E. Non-Normative Text.....	106
F. Revision History	107

1 Introduction

0 WS-Policy defines a framework for allowing web services to express their constraints and
1 requirements. Such constraints and requirements are expressed as policy assertions. This
2 document defines a set of security policy assertions for use with the [WS-Policy](#) framework
3 with respect to security features provided in [WSS: SOAP Message Security](#), [WS-Trust](#) and
4 [WS-SecureConversation](#). This document takes the approach of defining a base set of
5 assertions that describe how messages are to be secured. Flexibility with respect to token
6 types, cryptographic algorithms and mechanisms used, including using transport level
7 security is part of the design and allows for evolution over time. The intent is to provide
8 enough information for compatibility and interoperability to be determined by web service
9 participants along with all information necessary to actually enable a participant to engage
10 in a secure exchange of messages.

11
12 Section 11, all examples and all Appendices are non-normative.

13 1.1 Example

14 Table 1 shows an "Effective Policy" example, including binding assertions and associated
15 property assertions, token assertions and integrity and confidentiality assertions.

16 *Table 1: Example security policy.*

```
17 (01) <wsp:Policy>  
18 (02)   <sp:SymmetricBinding>  
19 (03)     <wsp:Policy>  
20 (04)       <sp:ProtectionToken>  
21 (05)         <wsp:Policy>  
22 (06)           <sp:KerberosV5APREQToken  
23 (07)             sp:IncludeToken=".../IncludeToken/Once" />  
24 (08)           </wsp:Policy>  
25 (09)         </sp:ProtectionToken>  
26 (10)       <sp:SignBeforeEncrypting />  
27 (11)       <sp:EncryptSignature />  
28 (12)     </wsp:Policy>  
29 (13)   </sp:SymmetricBinding>  
30 (14)   <sp:SignedParts>  
31 (15)     <sp:Body/>  
32 (16)     <sp:Header  
33 (17)       Namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"  
34 (18)     />  
35 (19)   </sp:SignedParts>  
36 (20)   <sp:EncryptedParts>  
37 (21)     <sp:Body/>  
38 (22)   </sp:EncryptedParts>  
39 (23) </wsp:Policy>  
40
```

41 Line 1 in Table 1 indicates that this is a policy statement and that all assertions contained
 42 by the `wsp:Policy` element are required to be satisfied. Line 2 indicates the kind of security
 43 binding in force. Line 3 indicates a nested `wsp:Policy` element which contains assertions
 44 that qualify the behavior of the SymmetricBinding assertion. Line 4 indicates a
 45 ProtectionToken assertion. Line 5 indicates a nested `wsp:Policy` element which contains
 46 assertions indicating the type of token to be used for the ProtectionToken. Line 6 indicates
 47 that a Kerberos V5 APREQ token is to be used by both parties in a message exchange for
 48 protection. Line 9 indicates that signatures are generated over plaintext rather than
 49 ciphertext. Line 10 indicates that the signature over the signed messages parts is required
 50 to be encrypted. Lines 13-16 indicate which message parts are to be covered by the primary
 51 signature; in this case the `soap:Body` element, indicated by Line 14 and any SOAP headers
 52 in the WS-Addressing namespace, indicated by line 15. Lines 17-19 indicate which message
 53 parts are to be encrypted; in this case just the `soap:Body` element, indicated by Line 18.

54 1.2 Namespaces

55 The XML namespace URI that MUST be used by implementations of this specification is:

56 `http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512`

57

58 Table 2 lists XML namespaces that are used in this specification. The choice of any
 59 namespace prefix is arbitrary and not semantically significant.

60 *Table 2: Prefixes and XML Namespaces used in this specification.*

Prefix	Namespace	Specification(s)
S	<code>http://schemas.xmlsoap.org/soap/envelope/</code>	[SOAP11]
ds	<code>http://www.w3.org/2000/09/xmldsig#</code>	[XMLDSIG]
enc	<code>http://www.w3.org/2001/04/xmlenc#</code>	[XMLENC]
wsu	<code>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd</code>	[WSS10]
wsse	<code>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd</code>	[WSS10]
wsse11	<code>http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-wss-wssecurity-secext-1.1.xsd</code>	[WSS11]
wsp	<code>http://schemas.xmlsoap.org/ws/2004/09/policy</code>	[WS-Policy], [WS-PolicyAttachment]
xsd	<code>http://www.w3.org/2001/XMLSchema</code>	[XMLSchema Part1], [XMLSchema Part2]
wst	<code>http://docs.oasis-open.org/ws-sx/ws-trust/200512</code>	[WS-Trust]
wsc	<code>http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512</code>	[WS-SecureConversation]
wsa	<code>http://schemas.xmlsoap.org/ws/2004/08/addressing</code>	[WS-Addressing]
sp	<code>http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512</code>	This specification

61 1.3 Schema Files

62 A normative copy of the XML Schema [XML Schema Part 1, Part 2] description for this
63 specification can be retrieved from the following address:

64 [http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/ws-
securitypolicy.xsd](http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/ws-
65 securitypolicy.xsd)

66 1.4 Terminology

67 **Policy** - A collection of policy alternatives.

68 **Policy Alternative** - A collection of policy assertions.

69 **Policy Assertion** - An individual requirement, capability, other property, or a behavior.

70 **Initiator** - The role sending the initial message in a message exchange.

71 **Recipient** - The targeted role to process the initial message in a message exchange.

72 **Security Binding** - A set of properties that together provide enough information to secure
73 a given message exchange.

74 **Security Binding Property** - A particular aspect of securing an exchange of messages.

75 **Security Binding Assertion** - A policy assertion that identifies the type of security binding
76 being used to secure an exchange of messages.

77 **Security Binding Property Assertion** - A policy assertion that specifies a particular value
78 for a particular aspect of securing an exchange of message.

79 **Assertion Parameter** - An element of variability within a policy assertion.

80 **Token Assertion** - Describes a token requirement. Token assertions defined within a
81 security binding are used to satisfy protection requirements.

82 **Supporting Token** - A token used to provide additional claims.

83 1.4.1 Notational Conventions

84 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",
85 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be
86 interpreted as described in [RFC2119].

87 This specification uses the following syntax to define outlines for assertions:

88 The syntax appears as an XML instance, but values in italics indicate data types
89 instead of literal values.

90 Characters are appended to elements and attributes to indicate cardinality:

- 91 ○ "?" (0 or 1)
- 92 ○ "*" (0 or more)
- 93 ○ "+" (1 or more)

94 The character "|" is used to indicate a choice between alternatives.

95 The characters "(" and ")" are used to indicate that contained items are to be treated
96 as a group with respect to cardinality or choice.

97 The characters "[" and "]" are used to call out references and property names.

98 Ellipses (i.e., "...") indicate points of extensibility. Additional children and/or
99 attributes MAY be added at the indicated extension points but MUST NOT contradict
100 the semantics of the parent and/or owner, respectively. By default, if a receiver does
101 not recognize an extension, the receiver SHOULD ignore the extension; exceptions to
102 this processing rule, if any, are clearly indicated below.

103 XML namespace prefixes (see Table 2) are used to indicate the namespace of the
104 element being defined.

105 In this document reference is made to the `wsu:Id` attribute and the `wsu:Created` and
106 `wsu:Expires` elements in a utility schema ([http://docs.oasis-open.org/wss/2004/01/oasis-
107 200401-wss-wssecurity-utility-1.0.xsd](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd)). The `wsu:Id` attribute and the `wsu:Created` and
108 `wsu:Expires` elements were added to the utility schema with the intent that other
109 specifications requiring such an ID or timestamp could reference it (as is done here).

110

111 WS-SecurityPolicy is designed to work with the general Web Services framework including
112 WSDL service descriptions, UDDI businessServices and bindingTemplates and SOAP
113 message structure and message processing model, and WS-SecurityPolicy should be
114 applicable to any version of SOAP. The current SOAP 1.2 namespace URI is used herein to
115 provide detailed examples, but there is no intention to limit the applicability of this
116 specification to a single version of SOAP.

117 1.5 Normative References

- 118 [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
119 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.
- 120 [KEYWORDS] S. Bradner, "Key words for use in RFCs to Indicate Requirement
121 Levels," RFC 2119, Harvard University, March 1997
- 122 [RFC2068] IETF Standard, "[Hypertext Transfer Protocol -- HTTP/1.1](#)" January
123 1997
- 124 [SOAP11] W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.
- 125 [SOAP12] W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging
126 Framework," 24 June 2003.
- 127 [XMLSchema Part1] W3C Recommendation, "[XML Schema Part 1: Structure Second
128 Edition](#)," 28 October 2004.
- 129 [XMLSchema Part2] W3C Recommendation, "[XML Schema Part 2: Datatypes Second
130 Edition](#)," 28 October 2004.
- 131 [URI] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers
132 (URI): Generic Syntax," RFC 3986, MIT/LCS, Day Software, Adobe
133 Systems, January 2005.

134 1.6 Non-Normative References

- 135 [WS-Policy] S.Bajaj, et.al., "[Web Services Policy Framework \(WS-Policy\)](#),"
136 September 2004
- 137 [WS-PolicyAttachment] S.Bajaj, et.al., "[Web Services Policy Attachment \(WS-
138 PolicyAttachment\)](#)," September 2004
- 139 [WS-Trust] S.Anderson, et.al., "[Web Services Trust Language \(WS-Trust\)](#),"
140 February 2005
- 141 [WS-SecureConversation] S.Anderson, et.al., "[Web Services Secure Conversation
142 Language \(WS-SecureConversation\)](#)," February 2005
- 143 [WS-Addressing] D.Box, et.al., "[Web Services Addressing Language \(WS-Addressing\)](#),"
144 August 2004
- 145 [WSS10] A. Nadalin, et.al., "[Web Services Security: SOAP Message Security 1.0
146 \(WS-Security 2004\)](#)," OASIS Standard 200401, March 2004
- 147 [WSS:UsernameToken 1.0] A. Nadalin, et.al., "[Web Services Security: UsernameToken
148 Profile 1.0](#)," OASIS Standard 200401, March 2004

149 [WSS:X509Token] P. Hallam-Baker, et.al., "[Web Services Security X.509](#)
150 [Certificate Token Profile](#)," OASIS Standard 200401, March 2004
151 [WSS:Kerberos Token 1.0] TBD – update
152 [XMLDSIG] D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon.
153 *XML-Signature Syntax and Processing*, W3C Recommendation, 12
154 February 2002. <http://www.w3.org/TR/xmlsig-core/>.
155 [XMLENC] W3C Recommendation, "[XML Encryption Syntax and Processing](#)," 10
156 December 2002.
157 [XPATH] James Clark, Steve DeRose. "[XML Path Language \(XPath\) Version 1.0](#)",
158 W3C Recommendation, 16 February 1999.
159

160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201

2 Security Policy Model

This specification defines policy assertions for the security properties for Web services. These assertions are primarily designed to represent the security characteristics defined in the [WSS: SOAP Message Security](#), WS-Trust and WS-SecureConversation specifications, but they can also be used for describing security requirements at a more general or transport-independent level.

The primary goal of this specification is to define an initial set of patterns or sets of assertions that represent common ways to describe how messages are secured on a communication path. The intent is to allow flexibility in terms of the tokens, cryptography, and mechanisms used, including leveraging transport security, but to be specific enough to ensure interoperability based on assertion matching.

It is a goal of the security policy model to leverage the WS-Policy framework's intersection algorithm for selecting policy alternatives and the attachment mechanism for associating policy assertions with web service artifacts. Consequently, wherever possible, the security policy assertions do not use parameters or attributes. This enables first-level, QName based assertion matching without security domain-specific knowledge to be done at the framework level. The first level matching is intended to provide a narrowed set of policy alternatives that are shared by the two parties attempting to establish a secure communication path.

In general, assertions defined in this specification allow additional attributes, based on schemas, to be added on to the assertion element as an extensibility mechanism but the WS-Policy framework will not match based on these attributes. Attributes specified on the assertion element that are not defined in this specification or in WS-Policy are to be treated as informational properties.

2.1 Security Assertion Model

The goal to provide richer semantics for combinations of security constraints and requirements and enable first-level QName matching, is enabled by the assertions defined in this specification being separated into simple patterns: what parts of a message are being secured (Protection Assertions), general aspects or pre-conditions of the security (Conditional Assertions), the security mechanism (Security Binding Assertions) that is used to provide the security, the token types and usage patterns (Supporting Token Assertions) used to provide additional claims, and token referencing and trust options (WSS and Trust Assertions).

To indicate the scope of protection, assertions identify message parts that are to be protected in a specific way, such as integrity or confidentiality protection, and are referred to as protection assertions.

The general aspects of security includes the relationships between or characteristics of the environment in which security is being applied, such as the tokens being used, which are for

202 integrity or confidentiality protection and which are supporting, the applicable algorithms to
203 use, etc.

204

205 The security binding assertion is a logical grouping which defines how the general aspects
206 are used to protect the indicated parts. For example, that an asymmetric token is used with
207 a digital signature to provide integrity protection, and that parts are encrypted with a
208 symmetric key which is then encrypted using the public key of the recipient. At its simplest
209 form, the security binding restricts what can be placed in the `wsse:Security` header and
210 the associated processing rules.

211

212 The intent of representing characteristics as assertions, is so that QName matching will be
213 sufficient to find common alternatives, and so that many aspects of security can be factored
214 out and re-used. For example, it may be common that the mechanism is constant for an
215 endpoint, but that the parts protected vary by message action.

216 **2.2 Nested Policy Assertions**

217 Assertions may be used to further qualify a specific aspect of another assertion. For
218 example, an assertion describing the set of algorithms to use may qualify the specific
219 behavior of a security binding. To enable this set of functionality, this specification
220 introduces a mechanism for nesting policy assertions underneath other assertions. This
221 mechanism is described in Section 3.

222 **2.3 Security Binding Abstraction**

223 As previously indicated, individual assertions are designed to be used in multiple
224 combinations. The binding represents common usage patterns for security mechanisms.
225 These Security Binding assertions are used to determine how the security is performed and
226 what to expect in the `wsse:Security` header.

227 Bindings are described textually and enforced programmatically. This specification defines
228 several bindings but others can be defined and agreed to for interoperability if participating
229 parties support it.

230

231 A binding defines the following security characteristics:

232 The minimum set of tokens that will be used and how they are bound to messages.
233 Note that services might accept messages containing more tokens than those
234 specified in policy.

235 Any necessary key transport mechanisms

236 Any required message elements (e.g. timestamps) in the `wsse:Security` header.

237 The content and ordering of elements in the `wsse:Security` header. Elements not
238 specified in the binding are not allowed.

239 Various parameters, including those describing the algorithms to be used for
240 canonicalization, signing and encryption.

241

242 Together the above pieces of information, along with the assertions describing conditions
243 and scope, provide enough information to secure messages between an initiator and a
244 recipient.

245

246 The following list identifies the bindings defined in this specification. The bindings are
247 identified primarily by the style of protection encryption used to protect the message
248 exchange. A later section of this document provides details on the assertions for these
249 bindings.

250 TransportBinding (Section 7.3)

251 SymmetricBinding (Section 7.4)

252 AsymmetricBinding (Section 7.5)

253 **3 Policy Considerations**

254 The following sections discuss details of WS-Policy and WS-PolicyAttachment relevant to this
255 specification.

256 **3.1 Nested Policy**

257 The WS-Policy specification defines a mechanism for describing capabilities and
258 requirements as assertions. These assertions are contained within one of `wsp:Policy`,
259 `wsp:All`, or `wsp:ExactlyOne`. The WS-Policy specification defines the nesting semantics
260 associated with the `wsp:Policy`, `wsp:All` and `wsp:ExactlyOne`. However these semantics do
261 not allow individual assertions to specify that the child elements contained within the
262 assertion should also be evaluated as assertions.

263
264 The following section is an overview of the nesting semantics of WS-Policy elements.

265 **3.1.1 Nesting Policy Elements**

266 To determine whether two assertions are "compatible", the QName value, that is the Name
267 and Namespace of one assertion element is compared to the QName value of another
268 assertion. If they match, then they are compatible.

269
270 A `wsp:Policy` element may contain one or more assertions. To determine whether two
271 `wsp:Policy` elements are "compatible", each assertion in one `wsp:Policy` element is
272 compared, as described above, to the assertions in another `wsp:Policy` element. If all
273 assertions from each `wsp:Policy` element are matched exactly, they are compatible.

274 To enable richer sets of options to be expressed, WS-Policy defines the `wsp:All` and
275 `wsp:ExactlyOne` elements. These elements may be placed as immediate children of a
276 `wsp:Policy` element. In addition, these two elements may also appear under themselves.
277 This allows for a policy to describe alternative options within policy. Let's say that a policy
278 wishes to express requirements for A and (B or C). This could be described as two policy
279 statements:

```
280 <wsp:Policy>  
281   <A />  
282   <B />  
283 </wsp:Policy>  
284 <wsp:Policy>  
285   <A />  
286   <C />  
287 </wsp:Policy>
```

288
289 Alternatively, we can use the `wsp:All` and `wsp:ExactlyOne` elements to describe the
290 alternative policy in a single `wsp:Policy` element:

```
291 <wsp:Policy>
292   <wsp:All>
293     <A />
294     <wsp:ExactlyOne>
295       <B />
296       <C />
297     </wsp:ExactlyOne>
298   </wsp:All>
299 </wsp:Policy>
```

300 This process is described in more detail in the WS-Policy specification.

301 **3.1.2 Nested Policy Assertions**

302 Some assertions may need to declare that additional assertions, scoped only to that
303 assertion, further qualify the behavior and compatibility semantics of that assertion.
304 Whereas the `wsp:All` and `wsp:ExactlyOne` elements describe requirements and alternatives
305 of a `wsp:Policy` element, nested assertions describe requirements and alternatives for the
306 enclosing assertion element. To enable these semantics, this specification defines some
307 assertions such that they have a single `wsp:Policy` child element which in turn contains
308 assertions which qualify the behavior of the enclosing assertion. Two such assertions are
309 compatible if they have the same QName AND their nested policy expressions (if any) are
310 compatible.

311
312 For example, let's say that a policy wishes to express requirements for A and B, and
313 furthermore that B requires C and D. The normalized policy statement would look like:

```
314 <wsp:Policy>
315   <A />
316   <B>
317     <wsp:Policy>
318       <C />
319       <D />
320     </wsp:Policy>
321   </B>
322 </wsp:Policy>
```

323 The policy above is fully normalized. Policy normalization DOES NOT promote nested
324 assertions to the outer scope.

325
326 The `wsp:Policy` element allows any assertion as content. However, assertions defined in this
327 specification that allow nested policy will typically constrain the content of that nested
328 policy.

329
330 Note: To enable automatic intersection of nested policy assertions, policy engines will need
331 to be modified to scan the contents of assertions to determine whether intersection is
332 required. This approach is being investigated by the WS-Policy working group to formalize
333 the notion of nested policy and to define processing behavior requirements for nested
334 policy. Additionally, an attribute may be defined to advertise to a policy engine that
335 scanning is required on a particular assertion. For example:

```
336 <wsp:Policy>
337   <A />
338   <B x:ContainsPolicy="true">
339     <wsp:Policy>
340       ...
341     </wsp:Policy>
342   </B>
343 </wsp:Policy>
```

344
345 Ideally the x:ContainsPolicy attribute will, at some point, be moved in the WS-Policy
346 namespace.

347 **3.1.3 Nesting Policy Processing Rules**

348 This section provides rules for processing nested policy based on the informal description
349 above;

- 350 1. Assertions MUST specify whether or not they contain nested policy.
- 351 2. Assertions SHOULD specify which other assertions can be present in their nested
352 policy.
- 353 3. Nested assertions need to be specifically designed for nesting inside one or more
354 outer assertions. Assertions SHOULD specify which assertions they can be nested
355 within.
- 356 4. Assertions from one domain SHOULD NOT be nested inside assertions from another
357 domain. For example, assertions from a transaction domain should not be nested
358 inside an assertion from a security domain.
- 359 5. Assertions containing nested policy are normalized recursively such that in the
360 normal form each nested policy contains no choices. Thus each outer assertion that
361 contains nested policy containing choices is duplicated such that there are as many
362 instances of the outer assertion as there are choices in the nested policy, one
363 instance for each nested choice, recursively. See Section 3.1.4 for a worked example
364 of normalization.
- 365 6. Nested policies are intersected in their own processing contexts with the
366 corresponding nested policy in a matching outer assertion. Thus two assertions
367 having nested policy intersect if the outer assertion QName matches and the nested
368 policies intersect. Intersection always occurs using the normalized form. See Section
369 3.1.5 for a worked example of intersection.
- 370 7. An assertion with an empty nested policy does not intersect with the same assertion
371 without nested policy.

372 **3.1.4 Nested Policy Normalization Worked Example**

373 This section shows a worked example of normalizing assertions with nested policy.
374


```

375 <wsp:Policy>
376   <wsp:ExactlyOne>
377     <wsp:All>
378       <A />
379       <B>
380         <wsp:Policy>
381           <wsp:ExactlyOne>
382             <wsp:All>
383               <C/>
384             </wsp:All>
385           <wsp:All>
386             <D/>
387           </wsp:All>
388         </wsp:ExactlyOne>
389       </wsp:Policy>
390     </B>
391   </wsp:All>
392 </wsp:ExactlyOne>
393 </wsp:Policy>

```

394
395 The above policy is normalized by, in this case, creating two alternatives, both containing an
396 A assertion and a B assertion. One alternative contains a B assertion with a nested C
397 assertion while the other contains a B assertion with a nested D assertion (normalized
398 form);
399

```

400 <wsp:Policy>
401   <wsp:ExactlyOne>
402     <wsp:All>
403       <A/>
404       <B>
405         <wsp:Policy>
406           <wsp:ExactlyOne>
407             <wsp:All>
408               <C/>
409             </wsp:All>
410           </wsp:ExactlyOne>
411         </wsp:Policy>
412       </B>
413     </wsp:All>
414   <wsp:All>
415     <A/>
416     <B>
417       <wsp:Policy>
418         <wsp:ExactlyOne>
419           <wsp:All>
420             <D/>
421           </wsp:All>
422         </wsp:ExactlyOne>
423       </wsp:Policy>
424     </B>
425   </wsp:All>
426 </wsp:ExactlyOne>
427 </wsp:Policy>

```

428 **3.1.5 Nested Policy Intersection Worked Example**

429 This section shows a worked example of computing the intersection of two policies that
430 contain assertions with nested policy.
431

```

432 <wsp:Policy>
433   <wsp:ExactlyOne>
434     <wsp:All>
435       <A />
436       <B>
437         <wsp:Policy>
438           <wsp:ExactlyOne>
439             <wsp:All>
440               <C/>
441             </wsp:All>
442           </wsp:ExactlyOne>
443         </wsp:Policy>
444       </B>
445     </wsp:All>
446   <wsp:All>
447     <A />
448     <B>
449       <wsp:Policy>
450         <wsp:ExactlyOne>
451           <wsp:All>
452             <D/>
453           </wsp:All>
454         </wsp:ExactlyOne>
455       </wsp:Policy>
456     </B>
457   </wsp:All>
458 </wsp:ExactlyOne>
459 </wsp:Policy>

```

```

460
461 <wsp:Policy>
462   <wsp:ExactlyOne>
463     <wsp:All>
464       <A />
465       <B>
466         <wsp:Policy>
467           <wsp:ExactlyOne>
468             <wsp:All>
469               <C/>
470             </wsp:All>
471           </wsp:ExactlyOne>
472         </wsp:Policy>
473       </B>
474     </wsp:All>
475   <wsp:All>
476     <A />
477     <B>
478       <wsp:Policy>
479         <wsp:ExactlyOne>
480           <wsp:All>
481             <E/>
482           </wsp:All>
483         </wsp:ExactlyOne>
484       </wsp:Policy>
485     </B>
486   </wsp:All>
487 </wsp:ExactlyOne>
488 </wsp:Policy>

```

489
490 The two policies above, which are already in normal form, are intersected as follows; firstly
491 the QNames of the A and B assertions are intersected then the QNames of the nested
492 assertions inside the B assertions are intersected. In the nested case, only the two B

493 assertions that have nested C assertions match. Thus the intersection of the nested policy is
494 (intersected policy;
495

```
496 <wsp:Policy>  
497   <wsp:ExactlyOne>  
498     <wsp:All>  
499       <A/>  
500       <A/>  
501       <B>  
502         <wsp:Policy>  
503           <wsp:ExactlyOne>  
504             <wsp:All>  
505               <C/>  
506             </wsp:All>  
507           </wsp:ExactlyOne>  
508         </wsp:Policy>  
509       </B>  
510     </wsp:All>  
511   </wsp:ExactlyOne>  
512 </wsp:Policy>  
513 </wsp:Policy>  
514 </wsp:Policy>  
515 </wsp:Policy>  
516 </wsp:Policy>  
517 </wsp:Policy>  
518 </wsp:Policy>  
519 </wsp:Policy>  
520 </wsp:Policy>  
521 </wsp:Policy>
```

522

523 **3.2 Policy Subjects**

524 WS-PolicyAttachment defines various attachment points for policy. This section defines
525 properties that are referenced later in this document describing the recommended or
526 required attachment points for various assertions. In addition, Appendix A groups the
527 various assertions according to policy subject.

528 Note: This specification does not define any assertions that have a scope of [Service Policy
529 Subject].

530 **[Message Policy Subject]**

531 This property identifies a Message Policy Subject [WS-PolicyAttachment]. WS-
532 PolicyAttachment defines seven WSDL [[WSDL 1.1](#)] policy attachment points with Message
533 Policy Subject:

534

535 wsdl:message

536 A policy expression containing one or more assertions with Message Policy Subject
537 MUST NOT be attached to a wsdl:message.

538 wsdl:portType/wsdl:operation/wsdl:input, ./wsdl:output, or ./wsdl:fault

539 A policy expression containing one or more assertions with Message Policy Subject
540 MUST NOT be attached to a descendant of wsdl:portType.

541 wsdl:binding/wsdl:operation/wsdl:input, ./wsdl:output, or ./wsdl:fault

542 A policy expression containing one or more of the assertions with Message Policy
543 Subject MUST be attached to a descendant of wsdl:binding.

544 **[Operation Policy Subject]**

545 A token assertion with Operation Policy Subject indicates usage of the token on a per-
546 operation basis:

547 wsdl:portType/wsdl:operation

548 A policy expression containing one or more token assertions MUST NOT be attached
549 to a wsdl:portType/wsdl:operation.

550 wsdl:binding/wsdl:operation

551 A policy expression containing one or
552 more token assertions MUST be attached to a wsdl:binding/wsdl:operation.

553

554 **[Endpoint Policy Subject]**

555 A token assertion instance with Endpoint Policy Subject indicates usage of the token for the
556 entire set of messages described for the endpoint:

557 wsdl:portType

558 A policy expression containing one or more assertions with Endpoint Policy Subject
559 MUST NOT be attached to a wsdl:portType.

560 wsdl:binding

561 A policy expression containing one or more of the assertions with Endpoint Policy
562 Subject SHOULD be attached to a wsdl:binding.

563 wsdl:port

564 A policy expression containing one or more of the assertions with Endpoint Policy
565 Subject MAY be attached to a wsdl:port

566 4 Protection Assertions

567 The following assertions are used to identify *what* is being protected and the level of
568 protection provided. These assertions SHOULD apply to [Message Policy Subject]. These
569 assertions MAY apply to [Endpoint Policy Subject] or [Operation Policy Subject]. Where they
570 apply to [Operation Policy Subject] they apply to all messages of that operation. Where they
571 apply to [Endpoint Policy Subject] they apply to all operations of that endpoint.

572 Note that when assertions defined in this section are present in a policy, the order of those
573 assertions in that policy has no effect on the order of signature and encryption operations
574 (see Section 6.3).

575 4.1 Integrity Assertions

576 Two mechanisms are defined for specifying the set of message parts to integrity protect.
577 One uses QNames to specify either message headers or the message body while the other
578 uses XPath expressions to identify any part of the message.

579 4.1.1 SignedParts Assertion

580 The SignedParts assertion is used to specify the parts of the message outside of security
581 headers that require integrity protection. This assertion can be satisfied using WSS: SOAP
582 Message Security mechanisms or by mechanisms out of scope of SOAP message security,
583 for example by sending the message over a secure transport protocol like HTTPS. The
584 binding details the exact mechanism by which the protection is provided.

585

586 There MAY be multiple SignedParts assertions present. Multiple SignedParts assertions
587 present within a policy alternative are equivalent to a single SignedParts assertion
588 containing the union of all specified message parts. Note that this assertion does not require
589 that a given part appear in a message, just that if such a part appears, it requires integrity
590 protection.

591 Syntax

```
592 <sp:SignedParts ... >  
593   <sp:Body />?  
594   <sp:Header Name="xs:NCName"? Namespace="xs:anyURI" ... />*  
595   ...  
596 </sp:SignedParts>
```

597

598 The following describes the attributes and elements listed in the schema outlined above:

599 /sp:SignedParts

600 This assertion specifies the parts of the message that need integrity protection. If no
601 child elements are specified, all message headers targeted at the UltimateReceiver
602 role [SOAP12] or actor [SOAP11] and the body of the message MUST be integrity
603 protected.

604 /sp:SignedParts/sp:Body

605 Presence of this optional empty element indicates that the entire body, that is the
606 soap:Body element, it's attributes and content, of the message needs to be integrity
607 protected.

608 /sp:SignedParts/sp:Header
609 Presence of this optional element indicates a specific SOAP header (or set of such
610 headers) needs to be protected. There may be multiple sp:Header elements within a
611 single sp:SignedParts element. If multiple SOAP headers with the same local name
612 but different namespace names are to be integrity protected multiple sp:Header
613 elements are needed, either as part of a single sp:SignedParts assertion or as part of
614 separate sp:SignedParts assertions.
615 This element only applies to SOAP header elements targeted to the same actor/role
616 as the Security header impacted by the policy. If it is necessary to specify a
617 requirement to sign specific SOAP Header elements targeted to a different actor/role,
618 that may be accomplished using the sp:SignedElements assertion.

619 /sp:SignedParts/sp:Header/@Name
620 This optional attribute indicates the local name of the SOAP header to be integrity
621 protected. If this attribute is not specified, all SOAP headers whose namespace
622 matches the Namespace attribute are to be protected.

623 /sp:SignedParts/sp:Header/@Namespace
624 This required attribute indicates the namespace of the SOAP header(s) to be integrity
625 protected.

626 4.1.2 SignedElements Assertion

627 The SignedElements assertion is used to specify arbitrary elements in the message that
628 require integrity protection. This assertion can be satisfied using WSS: SOAP Message
629 Security mechanisms or by mechanisms out of scope of SOAP message security, for
630 example by sending the message over a secure transport protocol like HTTPS. The binding
631 details the exact mechanism by which the protection is provided.

632

633 There MAY be multiple SignedElements assertions present. Multiple SignedElements
634 assertions present within a policy alternative are equivalent to a single SignedElements
635 assertion containing the union of all specified XPath expressions.

636 Syntax

```
637 <sp:SignedElements XPathVersion="xs:anyURI"? ... >  
638   <sp:XPath>xs:string</sp:XPath>+  
639   ...  
640 </sp:SignedElements>
```

641 The following describes the attributes and elements listed in the schema outlined above:

642 /sp:SignedElements

643 This assertion specifies the parts of the message that need integrity protection.

644 /sp:SignedElements/@XPathVersion

645 This optional attribute contains a URI which indicates the version of XPath to use.

646 /sp:SignedElements/sp:XPath

647 This element contains a string specifying an XPath expression that identifies the
648 nodes to be integrity protected. The XPath expression is evaluated against the
649 S:Envelope element node of the message. Multiple instances of this element may
650 appear within this assertion and should be treated as separate references in the
651 signature.

652 4.2 Confidentiality Assertions

653 Two mechanisms are defined for specifying the set of message parts to confidentiality
654 protect. One uses QNames to specify either message headers or the message body while
655 the other uses XPath expressions to identify any part of the message.

656 4.2.1 EncryptedParts Assertion

657 The EncryptedParts assertion is used to specify the parts of the message that require
658 confidentiality. This assertion can be satisfied with WSS: SOAP Message Security
659 mechanisms or by mechanisms out of scope of SOAP message security, for example by
660 sending the message over a secure transport protocol like HTTPS. The binding details the
661 exact mechanism by which the protection is provided.

662
663 There MAY be multiple EncryptedParts assertions present. Multiple EncryptedParts
664 assertions present within a policy alternative are equivalent to a single EncryptedParts
665 assertion containing the union of all specified message parts. Note that this assertion does
666 not require that a given part appear in a message, just that if such a part appears, it
667 requires confidentiality protection.

668 Syntax

```
669 <sp:EncryptedParts ... >  
670   <sp:Body/>?  
671   <sp:Header Name="xs:NCName"? Namespace="xs:anyURI" ... />*  
672   ...  
673 </sp:EncryptedParts>
```

674
675 The following describes the attributes and elements listed in the schema outlined above:
676 /sp:EncryptedParts

677 This assertion specifies the parts of the message that need confidentiality protection.
678 The single child element of this assertion specifies the set of message parts using an
679 extensible dialect.

680 If no child elements are specified, the body of the message MUST be confidentiality
681 protected.

682 /sp:EncryptedParts/sp:Body

683 Presence of this optional empty element indicates that the entire body of the
684 message needs to be confidentiality protected. In the case where mechanisms from
685 WSS: SOAP Message Security are used to satisfy this assertion, then the soap:Body
686 element is encrypted using the #Content encryption type.

687 /sp:EncryptedParts/sp:Header

688 Presence of this optional element indicates that a specific SOAP header (or set of
689 such headers) needs to be protected. There may be multiple sp:Header elements
690 within a single Parts element. Each header or set of headers MUST be encrypted.
691 Such encryption will encrypt such elements using WSS 1.1 Encrypted Headers. As
692 such, if WSS 1.1 Encrypted Headers are not supported by a service, then headers
693 cannot be encrypted using message level security. If multiple SOAP headers with the
694 same local name but different namespace names are to be encrypted then multiple
695 sp:Header elements are needed, either as part of a single sp:EncryptedParts
696 assertion or as part of separate sp:EncryptedParts assertions.

697 /sp:EncryptedParts/sp:Header/@Name
698 This optional attribute indicates the local name of the SOAP header to be
699 confidentiality protected. If this attribute is not specified, all SOAP headers whose
700 namespace matches the Namespace attribute are to be protected.
701 /sp:EncryptedParts/sp:Header/@Namespace
702 This required attribute indicates the namespace of the SOAP header(s) to be
703 confidentiality protected.

704 4.2.2 EncryptedElements Assertion

705 The EncryptedElements assertion is used to specify arbitrary elements in the message that
706 require confidentiality protection. This assertion can be satisfied using WSS: SOAP Message
707 Security mechanisms or by mechanisms out of scope of SOAP message security, for
708 example by sending the message over a secure transport protocol like HTTPS. The binding
709 details the exact mechanism by which the protection is provided.

710

711 There MAY be multiple EncryptedElements assertions present. Multiple EncryptedElements
712 assertions present within a policy alternative are equivalent to a single EncryptedElements
713 assertion containing the union of all specified XPath expressions.

714 Syntax

```
715 <sp:EncryptedElements XPathVersion="xs:anyURI"? ... >  
716 <sp:XPath>xs:string</sp:XPath>+  
717 ...  
718 </sp:EncryptedElements>
```

719 The following describes the attributes and elements listed in the schema outlined above:

720 /sp:EncryptedElements

721 This assertion specifies the parts of the message that need confidentiality protection.
722 Any such elements are subject to #Element encryption.

723 /sp:EncryptedElements/@XPathVersion

724 This optional attribute contains a URI which indicates the version of XPath to use.

725 /sp:EncryptedElements/sp:XPath

726 This element contains a string specifying an XPath expression that identifies the
727 nodes to be confidentiality protected. The XPath expression is evaluated against the
728 S:Envelope element node of the message. Multiple instances of this element may
729 appear within this assertion and should be treated as separate references.

730 4.3 Required Elements Assertion

731 A mechanism is defined for specifying, using XPath expressions, the set of header elements
732 that a message MUST contain.

733

734 Note: Specifications are expected to provide domain specific assertions that specify which
735 headers are expected in a message. This assertion is provided for cases where such domain
736 specific assertions have not been defined.

737 **4.3.1 RequiredElements Assertion**

738 The RequiredElements assertion is used to specify header elements that the message MUST
739 contain. This assertion specifies no security requirements.

740

741 There MAY be multiple RequiredElements assertions present. Multiple RequiredElements
742 assertions present within a policy alternative are equivalent to a single RequiredElements
743 assertion containing the union of all specified XPath expressions.

744 **Syntax**

```
745 <sp:RequiredElements XPathVersion="xs:anyURI"? ... >  
746   <sp:XPath>xs:string</sp:XPath>+  
747   ...  
748 </sp:RequiredElements>
```

749

750 The following describes the attributes and elements listed in the schema outlined above:

751 /sp:RequiredElements

752 This assertion specifies the headers elements that MUST appear in a message.

753 /sp:RequiredElements/@XPathVersion

754 This optional attribute contains a URI which indicates the version of XPath to use.

755 /sp:RequiredElements/sp:XPath

756 This element contains a string specifying an XPath expression that identifies the
757 header elements that a message MUST contain. The XPath expression is evaluated
758 against the S:Envelope/S:Header element node of the message. Multiple instances of
759 this element may appear within this assertion and should be treated as a combined
760 XPath expression.

761

5 Token Assertions

762

Token assertions specify the type of tokens to use to protect or bind tokens and claims to the message. These assertions do not recommend usage of a Policy Subject. Assertions which contain them SHOULD recommend a policy attachment point. With the exception of transport token assertions, the token assertions defined in this section are not specific to any particular security binding.

763

764

765

766

767

5.1 Token Inclusion

768

Any token assertion may also carry an optional `sp:IncludeToken` attribute. The schema type of this attribute is `xs:anyURI`. This attribute indicates whether the token should be included, that is written, in the message or whether cryptographic operations utilize an external reference mechanism to refer to the key represented by the token. This attribute is defined as a global attribute in the WS-SecurityPolicy namespace and is intended to be used by any specification that defines token assertions.

769

770

771

772

773

774

5.1.1 Token Inclusion Values

775

The following table describes the set of valid token inclusion mechanisms supported by this specification:

776

http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Never	The token MUST NOT be included in any messages sent between the initiator and the recipient; rather, an external reference to the token should be used.
http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Once	The token MUST be included in only one message sent from the initiator to the recipient. References to the token MAY use an internal reference mechanism. Subsequent related messages sent between the recipient and the initiator may refer to the token using an external reference mechanism.
http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/AlwaysToRecipient	The token MUST be included in all messages sent from initiator to the recipient. The token MUST NOT be included in messages sent from the recipient to the initiator.
http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/AlwaysToInitiator	The token MUST be included in all messages sent from the recipient to the initiator. The token MUST NOT be included in messages sent from the initiator to the recipient.
http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/IncludeToken/Always	The token MUST be included in all messages sent between the initiator and the recipient. This is the default behavior.

777

778

Note: In examples, the namespace URI is replaced with "...". For example, `.../IncludeToken/Never` is actually `http://docs.oasis-open.org/ws-sx/ws-`

779

780 securitypolicy/200512/IncludeToken/Never. Other token inclusion URI values MAY be
781 defined but are out-of-scope of this specification.

782 The default behavior characteristics defined by this specification if this attribute is not
783 specified on a token assertion are .../IncludeToken/Always.

784 **5.1.2 Token Inclusion and Token References**

785 A token assertion may carry a sp:IncludeToken attribute that requires that the token be
786 included in the message. The Web Services Security specifications [WSS10, WSS11] define
787 mechanisms for how tokens are included in a message.

788 Several Token assertions (see Section 5.3) support mechanisms for referencing tokens in
789 addition to Direct References, for example external URI references or references using a
790 Thumbprint.

791 Certain combination of sp:IncludeToken value and token reference assertions can result in a
792 token appearing in a message more than once. For example, if a token assertion carries a
793 sp:IncludeToken attribute with a value of '.../Always' and that token assertion also contains
794 a nested sp:RequireEmbeddedTokenReference (see Section 5.3.3) assertion, then the token
795 would be included twice in the message. While such combinations are not in error, they are
796 probably best avoided for efficiency reasons.

797 If a token assertion contains multiple reference assertions then references to that token are
798 required to contain all the specified reference types. For example, if a token assertion
799 contains nested sp:RequireIssuerSerialReference and sp:RequireThumbprintReference
800 assertions then references to that token contain both reference forms. Again, while such
801 combinations are not in error, they are probably best avoided for efficiency reasons.

802 **5.2 Token Properties**

803 **5.2.1 [Derived Keys] Property**

804 This boolean property specifies whether derived keys should be used as defined in WS-
805 SecureConversation. If the value is 'true', derived keys MUST be used. If the value is 'false',
806 derived keys MUST NOT be used. The value of this property applies to a specific token. The
807 value of this property is populated by assertions specific to the token. The default value for
808 this property is 'false'.

809 See the [Explicit Derived Keys] and [Implicit Derived Key] properties below for information
810 on how particular forms of derived keys are specified.

811 Where the key material associated with a token is asymmetric, this property applies to the
812 use of symmetric keys encrypted with the key material associated with the token.

813 **5.2.2 [Explicit Derived Keys] Property**

814 This boolean property specifies whether Explicit Derived Keys (see Section 7 of [WS-
815 SecureConversation]) are allowed. If the value is 'true' then Explicit Derived Keys MAY be
816 used. If the value is 'false' then Explicit Derived Keys MUST NOT be used.

817 **5.2.3 [Implicit Derived Keys] Property**

818 This boolean property specifies whether Implicit Derived Keys (see Section 7.3 of [WS-
819 SecureConversation]) are allowed. If the value is 'true' then Implicit Derived Keys MAY be
820 used. If the value is 'false' then Implicit Derived Keys MUST NOT be used.

821 **5.3 Token Assertion Types**

822 The following sections describe the token assertions defined as part of this specification.

823 **5.3.1 UsernameToken Assertion**

824 This element represents a requirement to include a username token.

825 **Syntax**

```
826 <sp:UsernameToken sp:IncludeToken="xs:anyURI"? ... >
827   <wsp:Policy>
828     (
829       <sp:NoPassword ... /> |
830       <sp:HashPassword ... />
831     ) ?
832     (
833       <sp:RequireDerivedKeys /> |
834       <sp:RequireImplicitDerivedKeys ... /> |
835       <sp:RequireExplicitDerivedKeys ... />
836     ) ?
837     (
838       <sp:WssUsernameToken10 ... /> |
839       <sp:WssUsernameToken11 ... />
840     ) ?
841     ...
842   </wsp:Policy> ?
843   ...
844 </sp:UsernameToken>
```

845
846 The following describes the attributes and elements listed in the schema outlined above:

847 /sp:UsernameToken

848 This identifies a UsernameToken assertion.

849 /sp:UsernameToken/@sp:IncludeToken

850 This optional attribute identifies the token inclusion value for this token assertion.

851 /sp:UsernameToken/wsp:Policy

852 This optional element identifies additional requirements for use of the
853 sp:UsernameToken assertion.

854 /sp:UsernameToken/wsp:Policy/sp:NoPassword

855 This optional element indicates that the wsse:Password element MUST NOT be
856 present in the Username token.

857 /sp:UsernameToken/wsp:Policy/sp:HashPassword

858 This optional element indicates that the wsse:Password element MUST be present in
859 the Username token and that the content of the wsse:Password element MUST
860 contain a hash of the timestamp, nonce and password as defined in [WSS: Username
861 Token Profile].

862 /sp:UsernameToken/wsp:Policy/sp:RequireDerivedKeys

863 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
864 Derived Keys] properties for this token to 'true'.

865 /sp:UsernameToken/wsp:Policy/sp:RequireExplicitDerivedKeys

866 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
867 for this token to 'true' and the [Implicit Derived Keys] property for this token to
868 'false'.

869 /sp:UsernameToken/wsp:Policy/sp:RequireImplicitDerivedKeys

870 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
871 for this token to 'true' and the [Explicit Derived Keys] property for this token to
872 'false'.

873 /sp:UsernameToken/wsp:Policy/sp:WssUsernameToken10

874 This optional element indicates that a Username token should be used as defined in
875 [WSS: Username Token Profile 1.0].

876 /sp:UsernameToken/wsp:Policy/sp:WssUsernameToken11

877 This optional element indicates that a Username token should be used as defined in
878 [WSS: Username Token Profile 1.1].

879 5.3.2 IssuedToken Assertion

880 This element represents a requirement for an issued token, that is one issued by some
881 token issuer using the mechanisms defined in WS-Trust. This assertion is used in 3rd party
882 scenarios. For example, the initiator may need to request a SAML token from a given token
883 issuer in order to secure messages sent to the recipient.

884 Syntax

```
885 <sp:IssuedToken sp:IncludeToken="xs:anyURI"? ... >  
886 <sp:Issuer>wsa:EndpointReferenceType</sp:Issuer?>  
887 <sp:RequestSecurityTokenTemplate TrustVersion="xs:anyURI"? >  
888 ...  
889 </sp:RequestSecurityTokenTemplate>  
890 <wsp:Policy>  
891 (  
892 <sp:RequireDerivedKeys ... /> |  
893 <sp:RequireImplicitDerivedKeys ... /> |  
894 <sp:RequireExplicitDerivedKeys ... />  
895 ) ?  
896 <sp:RequireExternalReference ... /> ?  
897 <sp:RequireInternalReference ... /> ?  
898 ...  
899 </wsp:Policy> ?  
900 ...  
901 </sp:IssuedToken>
```

902 The following describes the attributes and elements listed in the schema outlined above:

903 /sp:IssuedToken

904 This identifies an IssuedToken assertion.

905 /sp:IssuedToken/@sp:IncludeToken

906 This optional attribute identifies the token inclusion value for this token assertion.

907 /sp:IssuedToken/sp:Issuer

908 This optional element, of type wsa:EndpointReferenceType, contains a reference to the
909 issuer for the issued token.

910 /sp:IssuedToken/sp:RequestSecurityTokenTemplate

911 This required element contains elements which MUST be copied into the request sent to
912 the specified issuer. Note: the initiator is not required to understand the contents of this
913 element.

914 See Appendix B for details of the content of this element.

915 `/sp:IssuedToken/sp:RequestSecurityTokenTemplate/@TrustVersion`

916 This optional attribute contains a URI identifying the version of WS-Trust referenced by
917 the contents of this element.

918 `/sp:IssuedToken/wsp:Policy`

919 This optional element identifies additional requirements for use of the
920 `sp:IssuedToken` assertion.

921 `/sp:IssuedToken/wsp:Policy/sp:RequireDerivedKeys`

922 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
923 Derived Keys] properties for this token to 'true'.

924 `/sp:IssuedToken/wsp:Policy/sp:RequireExplicitDerivedKeys`

925 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
926 for this token to 'true' and the [Implicit Derived Keys] property for this token to
927 'false'.

928 `/sp:IssuedToken/wsp:Policy/sp:RequireImplicitDerivedKeys`

929 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
930 for this token to 'true' and the [Explicit Derived Keys] property for this token to
931 'false'.

932 `/sp:IssuedToken/wsp:Policy/sp:RequireInternalReference`

933 This optional element indicates whether an internal reference is required when
934 referencing this token.

935 Note: This reference will be supplied by the issuer of the token.

936 `/sp:IssuedToken/wsp:Policy/sp:RequireExternalReference`

937 This optional element indicates whether an external reference is required when
938 referencing this token.

939 Note: This reference will be supplied by the issuer of the token.

940 Note: The `IssuedToken` may or may not be associated with key material and such key
941 material may be symmetric or asymmetric. The `Binding` assertion will imply the type of key
942 associated with this token. Services may also include information in the
943 `sp:RequestSecurityTokenTemplate` element to explicitly define the expected key type. See
944 Appendix B for details of the `sp:RequestSecurityTokenTemplate` element.

945 **5.3.3 X509Token Assertion**

946 This element represents a requirement for a binary security token carrying an X509 token.

947 **Syntax**

```

948 <sp:X509Token sp:IncludeToken="xs:anyURI"? ... >
949   <wsp:Policy>
950     (
951       <sp:RequireDerivedKeys ... /> |
952       <sp:RequireExplicitDerivedKeys ... /> |
953       <sp:RequireImplicitDerivedKeys ... />
954     ) ?
955     <sp:RequireKeyIdentifierReference ... /> ?
956     <sp:RequireIssuerSerialReference ... /> ?
957     <sp:RequireEmbeddedTokenReference ... /> ?
958     <sp:RequireThumbprintReference ... /> ?
959     (
960       <sp:WssX509V3Token10 ... /> |
961       <sp:WssX509Pkcs7Token10 ... /> |
962       <sp:WssX509PkiPathV1Token10 ... /> |
963       <sp:WssX509V1Token11 ... /> |
964       <sp:WssX509V3Token11 ... /> |
965       <sp:WssX509Pkcs7Token11 ... /> |
966       <sp:WssX509PkiPathV1Token11 ... />
967     ) ?
968     ...
969   </wsp:Policy> ?
970   ...
971 </sp:X509Token>

```

972
973 The following describes the attributes and elements listed in the schema outlined above:
974 /sp:X509Token

975 This identifies an X509Token assertion.

976 /sp:X509Token/@sp:IncludeToken

977 This optional attribute identifies the token inclusion value for this token assertion.

978 /sp:X509Token/wsp:Policy

979 This optional element identifies additional requirements for use of the sp:X509Token
980 assertion.

981 /sp:X509Token/wsp:Policy/sp:RequireDerivedKeys

982 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
983 Derived Keys] properties for this token to 'true'.

984 /sp:X509Token/wsp:Policy/sp:RequireExplicitDerivedKeys

985 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
986 for this token to 'true' and the [Implicit Derived Keys] property for this token to
987 'false'.

988 /sp:X509Token/wsp:Policy/sp:RequireImplicitDerivedKeys

989 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
990 for this token to 'true' and the [Explicit Derived Keys] property for this token to
991 'false'.

992 /sp:X509Token/wsp:Policy/sp:RequireKeyIdentifierReference

993 This optional element indicates that a key identifier reference is required when
994 referencing this token.

995 /sp:X509Token/wsp:Policy/sp:RequireIssuerSerialReference

996 This optional element indicates that an issuer serial reference is required when
997 referencing this token.

998 /sp:X509Token/wsp:Policy/sp:RequireEmbeddedTokenReference
 999 This optional element indicates that an embedded token reference is required when
 1000 referencing this token.

1001 /sp:X509Token/wsp:Policy/sp:RequireThumbprintReference
 1002 This optional element indicates that a thumbprint reference is required when
 1003 referencing this token.

1004 /sp:X509Token/wsp:Policy/sp:WssX509V3Token10
 1005 This optional element indicates that an X509 Version 3 token should be used as
 1006 defined in [WSS: X509 Token Profile 1.0].

1007 /sp:X509Token/wsp:Policy/sp:WssX509Pkcs7Token10
 1008 This optional element indicates that an X509 PKCS7 token should be used as defined
 1009 in [WSS: X509 Token Profile 1.0].

1010 /sp:X509Token/wsp:Policy/sp:WssX509PkiPathV1Token10
 1011 This optional element indicates that an X509 PKI Path Version 1 token should be
 1012 used as defined in [WSS: X509 Token Profile 1.0].

1013 /sp:X509Token/wsp:Policy/sp:WssX509V1Token11
 1014 This optional element indicates that an X509 Version 1 token should be used as
 1015 defined in [WSS: X509 Token Profile 1.1].

1016 /sp:X509Token/wsp:Policy/sp:WssX509V3Token11
 1017 This optional element indicates that an X509 Version 3 token should be used as
 1018 defined in [WSS: X509 Token Profile 1.1].

1019 /sp:X509Token/wsp:Policy/sp:WssX509Pkcs7Token11
 1020 This optional element indicates that an X509 PKCS7 token should be used as defined
 1021 in [WSS: X509 Token Profile 1.1].

1022 /sp:X509Token/wsp:Policy/sp:WssX509PkiPathV1Token11
 1023 This optional element indicates that an X509 PKI Path Version 1 token should be
 1024 used as defined in [WSS: X509 Token Profile 1.1].

1025 **5.3.4 KerberosToken Assertion**

1026 This element represents a requirement for a Kerberos token.

1027 **Syntax**

```

1028 <sp:KerberosToken sp:IncludeToken="xs:anyURI"? ... >
1029   <wsp:Policy>
1030     (
1031       <sp:RequireDerivedKeys ... /> |
1032       <sp:RequireImplicitDerivedKeys ... /> |
1033       <sp:RequireExplicitDerivedKeys ... />
1034     ) ?
1035     <sp:RequireKeyIdentifierReference ... /> ?
1036     (
1037       <sp:WssKerberosV5ApReqToken11 ... /> |
1038       <sp:WssGssKerberosV5ApReqToken11 ... />
1039     ) ?
1040

```



```
1041     ...
1042     </wsp:Policy> ?
1043     ...
1044 </sp:KerberosToken>
```

1045
1046 The following describes the attributes and elements listed in the schema outlined above:

1047 /sp:KerberosToken

1048 This identifies a KerberosV5ApReqToken assertion.

1049 /sp:KerberosToken/@sp:IncludeToken

1050 This optional attribute identifies the token inclusion value for this token assertion.

1051 /sp:KerberosToken/wsp:Policy

1052 This optional element identifies additional requirements for use of the
1053 sp:KerberosToken assertion.

1054 /sp:KerberosToken/wsp:Policy/sp:RequireDerivedKeys

1055 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
1056 Derived Keys] properties for this token to 'true'.

1057 /sp:KerberosToken/wsp:Policy/sp:RequireExplicitDerivedKeys

1058 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
1059 for this token to 'true' and the [Implicit Derived Keys] property for this token to
1060 'false'.

1061 /sp:KerberosToken/wsp:Policy/sp:RequireImplicitDerivedKeys

1062 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
1063 for this token to 'true' and the [Explicit Derived Keys] property for this token to
1064 'false'.

1065 /sp:KerberosToken/wsp:Policy/sp:RequireKeyIdentifierReference

1066 This optional element indicates that a key identifier reference is required when
1067 referencing this token.

1068 /sp:KerberosToken/wsp:Policy/sp:WssKerberosV5ApReqToken11

1069 This optional element indicates that a Kerberos Version 5 AP-REQ token should be
1070 used as defined in [WSS: Kerberos Token Profile 1.1].

1071 /sp:KerberosToken/wsp:Policy/sp:WssGssKerberosV5ApReqToken11

1072 This optional element indicates that a GSS Kerberos Version 5 AP-REQ token should
1073 be used as defined in [WSS: Kerberos Token Profile 1.1].

1074 **5.3.5 SpnegoContextToken Assertion**

1075 This element represents a requirement for a SecurityContextToken obtained by executing an
1076 n-leg RST/RSTR SPNEGO binary negotiation protocol with the Web Service, as defined in
1077 WS-Trust.

1078 **Syntax**

```

1079 <sp:SpnegoContextToken sp:IncludeToken="xs:anyURI"? ... >
1080   <sp:Issuer>wsa:EndpointReferenceType</sp:Issuer> ?
1081   <wsp:Policy>
1082     (
1083       <sp:RequireDerivedKeys ... /> |
1084       <sp:RequireImplicitDerivedKeys ... /> |
1085       <sp:RequireExplicitDerivedKeys ... />
1086     ) ?
1087     ...
1088   </wsp:Policy> ?
1089   ...
1090 </sp:SpnegoContextToken>

```

1091

1092 The following describes the attributes and elements listed in the schema outlined above:

1093 /sp:SpnegoContextToken

1094 This identifies a SpnegoContextToken assertion.

1095 /sp:SpnegoContextToken/@sp:IncludeToken

1096 This optional attribute identifies the token inclusion value for this token assertion.

1097 /sp:SpnegoContextToken/sp:Issuer

1098 This optional element, of type `wsa:EndpointReferenceType`, contains a reference to the
 1099 issuer for the Spnego Context Token.

1100 /sp:SpnegoContextToken/wsp:Policy

1101 This optional element identifies additional requirements for use of the
 1102 sp:SpnegoContextToken assertion.

1103 /sp:SpnegoContextToken/wsp:Policy/sp:RequireDerivedKeys

1104 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
 1105 Derived Keys] properties for this token to 'true'.

1106 /sp:SpnegoContextToken/wsp:Policy/sp:RequireExplicitDerivedKeys

1107 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
 1108 for this token to 'true' and the [Implicit Derived Keys] property for this token to
 1109 'false'.

1110 /sp:SpnegoContextToken/wsp:Policy/sp:RequireImplicitDerivedKeys

1111 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
 1112 for this token to 'true' and the [Explicit Derived Keys] property for this token to
 1113 'false'.

1114 **5.3.6 SecurityContextToken Assertion**

1115 This element represents a requirement for a SecurityContextToken token.

1116 **Syntax**

```

1117 <sp:SecurityContextToken sp:IncludeToken="xs:anyURI"? ... >
1118   <wsp:Policy>
1119     (
1120       <sp:RequireDerivedKeys ... /> |
1121       <sp:RequireImplicitDerivedKeys ... /> |
1122       <sp:RequireExplicitDerivedKeys ... />
1123     ) ?
1124     <sp:RequireExternalUriReference ... /> ?
1125     <sp:SC200502SecurityContextToken ... /> ?
1126     ...
1127   </wsp:Policy> ?
1128   ...
1129 </sp:SecurityContextToken>

```

1130

1131 The following describes the attributes and elements listed in the schema outlined above:

1132 /sp:SecurityContextToken

1133 This identifies a SecurityContextToken assertion.

1134 /sp:SecurityContextToken/@sp:IncludeToken

1135 This optional attribute identifies the token inclusion value for this token assertion.

1136 /sp:SecurityContextToken/wsp:Policy

1137 This optional element identifies additional requirements for use of the
 1138 sp:SecurityContextToken assertion.

1139 /sp:SecurityContextToken/wsp:Policy/sp:RequireDerivedKeys

1140 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
 1141 Derived Keys] properties for this token to 'true'.

1142 /sp:SecurityContextToken/wsp:Policy/sp:RequireExplicitDerivedKeys

1143 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
 1144 for this token to 'true' and the [Implicit Derived Keys] property for this token to
 1145 'false'.

1146 /sp:SecurityContextToken/wsp:Policy/sp:RequireImplicitDerivedKeys

1147 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
 1148 for this token to 'true' and the [Explicit Derived Keys] property for this token to
 1149 'false'.

1150 /sp:SecurityContextToken/wsp:Policy/sp:RequireExternalUriReference

1151 This optional element indicates that an external URI reference is required when
 1152 referencing this token.

1153 /sp:SecurityContextToken/wsp:Policy/sp:SC200502SecurityContextToken

1154 This optional element indicates that a Security Context Token should be used as
 1155 defined in [WS-SecureConversation].

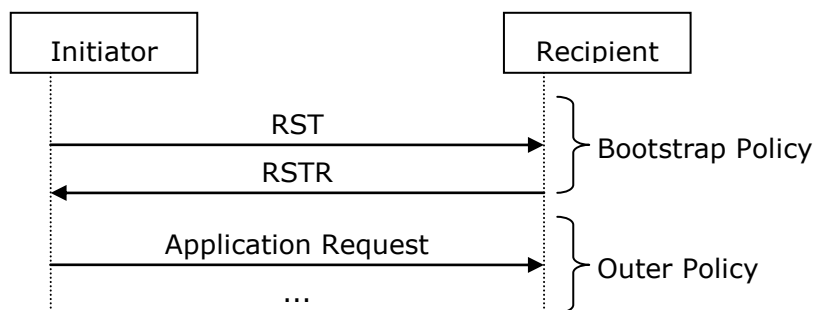
1156

1157 Note: This assertion does not describe how to obtain a Security Context Token but rather
 1158 assumes that both parties have the token already or have agreed separately on a
 1159 mechanism for obtaining the token. If a definition of the mechanism for obtaining the
 1160 Security Context Token is desired in policy, then either the sp:SecureConversationToken or
 1161 the sp:IssuedToken assertion should be used instead.

1162 5.3.7 SecureConversationToken Assertion

1163 This element represents a requirement for a Security Context Token retrieved from the
1164 indicated issuer address. If the sp:Issuer address is absent, the protocol MUST be executed
1165 at the same address as the service endpoint address.

1166
1167 Note: This assertion describes the token accepted by the target service. Because this token
1168 is issued by the target service and may not have a separate port (with separate policy), this
1169 assertion SHOULD contain a bootstrap policy indicating the security binding and policy that
1170 is used when requesting this token from the target service. That is, the bootstrap policy is
1171 used to obtain the token and then the current (outer) policy is used when making requests
1172 with the token. This is illustrated in the diagram below.



1173

1174 Syntax

```
1175 <sp:SecureConversationToken sp:IncludeToken="xs:anyURI"? ... >
1176   <sp:Issuer>wsa:EndpointReferenceType</sp:Issuer>?
1177   <wsp:Policy>
1178     (
1179       <sp:RequireDerivedKeys ... /> |
1180       <sp:RequireImplicitDerivedKeys ... /> |
1181       <sp:RequireExplicitDerivedKeys ... />
1182     ) ?
1183   <sp:RequireExternalUriReference ... /> ?
1184   <sp:SC200502SecurityContextToken ... /> ?
1185   <sp:BootstrapPolicy ... > ?
1186     <wsp:Policy> ... </wsp:Policy>
1187   </sp:BootstrapPolicy>
1188 </wsp:Policy> ?
1189   ...
1190 </sp:SecureConversationToken>
```

1191

1192 The following describes the attributes and elements listed in the schema outlined above:

1193 /sp:SecureConversationToken

1194 This identifies a SecureConversationToken assertion.

1195 /sp:SecureConversationToken/@sp:IncludeToken

1196 This optional attribute identifies the token inclusion value for this token assertion.

1197 /sp:SecureConversationToken/sp:Issuer

1198 This optional element, of type wsa:EndpointReferenceType, contains a reference to the
1199 issuer for the Security Context Token.

1200 /sp:SecureConversationToken/wsp:Policy

- 1201 This optional element identifies additional requirements for use of the
1202 sp:SecureConversationToken assertion.
- 1203 /sp:SecureConversationToken/wsp:Policy/sp:RequireDerivedKeys
- 1204 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
1205 Derived Keys] properties for this token to 'true'.
- 1206 /sp:SecureConversationToken/wsp:Policy/sp:RequireExplicitDerivedKeys
- 1207 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
1208 for this token to 'true' and the [Implicit Derived Keys] property for this token to
1209 'false'.
- 1210 /sp:SecureConversationToken/wsp:Policy/sp:RequireImplicitDerivedKeys
- 1211 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
1212 for this token to 'true' and the [Explicit Derived Keys] property for this token to
1213 'false'.
- 1214 /sp:SecureConversationToken/wsp:Policy/sp:RequireExternalUriReference
- 1215 This optional element indicates that an external URI reference is required when
1216 referencing this token.
- 1217 /sp:SecureConversationToken/wsp:Policy/sp:SC200502SecurityContextToken
- 1218 This optional element indicates that a Security Context Token should be used as
1219 obtained using the protocol defined in [WS-SecureConversation].
- 1220 /sp:SecureConversationToken/wsp:Policy/sp:BootstrapPolicy
- 1221 This optional element contains the policy indicating the requirements for obtaining the
1222 Security Context Token.
- 1223 /sp:SecureConversationToken/wsp:Policy/sp:BootstrapPolicy/wsp:Policy
- 1224 This element contains the security binding requirements for obtaining the Security
1225 Context Token. It will typically contain a security binding assertion (e.g.
1226 sp:SymmetricBinding) along with protection assertions (e.g. sp:SignedParts) describing
1227 the parts of the RST/RSTR messages that are to be protected.

1228 **Example**

1229
1230
1231
1232
1233
1234
1235
1236
1237
1238

```
<wsp:Policy>  
  <sp:SymmetricBinding>  
    <wsp:Policy>  
      <sp:ProtectionToken>  
        <wsp:Policy>  
          <sp:SecureConversationToken>  
            <sp:Issuer>  
              <wsa:Address>http://example.org/sts</wsa:Address>  
            </sp:Issuer>  
          </wsp:Policy>  
        </sp:ProtectionToken>  
      </wsp:Policy>  
    </sp:SymmetricBinding>  
  </wsp:Policy>
```

```

1239     <sp:SC10SecurityContextToken />
1240     <sp:BootstrapPolicy>
1241       <wsp:Policy>
1242         <sp:AsymmetricBinding>
1243           <wsp:Policy>
1244             <sp:InitiatorToken>
1245               ...
1246             </sp:InitiatorToken>
1247             <sp:RecipientToken>
1248               ...
1249             </sp:RecipientToken>
1250           </wsp:Policy>
1251         </sp:AsymmetricBinding>
1252         <sp:SignedParts>
1253           ...
1254         </sp:SignedParts>
1255       </wsp:Policy>
1256     </sp:BootstrapPolicy>
1257   </wsp:Policy>
1258 </sp:SecureConversationToken>
1259 </wsp:Policy>
1260 </sp:ProtectionToken>
1261 ...
1262 </wsp:Policy>
1263 </sp:SymmetricBinding>
1264 <sp:SignedParts>
1265 ...
1266 </sp:SignedParts>
1267 ...
1268 </wsp:Policy>
1269 </sp:Policy>

```

1270 5.3.8 SamlToken Assertion

1271 This element represents a requirement for a SAML token.

1272 Syntax

```

1273 <sp:SamlToken sp:IncludeToken="xs:anyURI"? ... >
1274   <wsp:Policy>
1275     (
1276       <sp:RequireDerivedKeys ... /> |
1277       <sp:RequireImplicitDerivedKeys ... /> |
1278       <sp:RequireExplicitDerivedKeys ... />
1279     ) ?
1280     <sp:RequireKeyIdentifierReference ... /> ?
1281     (
1282       <sp:WssSamlV11Token10 ... /> |
1283       <sp:WssSamlV11Token11 ... /> |
1284       <sp:WssSamlV20Token11 ... />
1285     ) ?
1286     ...
1287   </wsp:Policy> ?
1288   ...
1289 </sp:SamlToken>

```

1290
1291 The following describes the attributes and elements listed in the schema outlined above:

1292 /sp:SamlToken

1293 This identifies a SamlToken assertion.

1294 /sp:SamlToken/@sp:IncludeToken

1295 This optional attribute identifies the token inclusion value for this token assertion.
1296 /sp:SamlToken/wsp:Policy
1297 This optional element identifies additional requirements for use of the sp:SamlToken
1298 assertion.
1299 /sp:SamlToken/wsp:Policy/sp:RequireDerivedKeys
1300 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
1301 Derived Keys] properties for this token to 'true'.
1302 /sp:SamlToken/wsp:Policy/sp:RequireExplicitDerivedKeys
1303 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
1304 for this token to 'true' and the [Implicit Derived Keys] property for this token to
1305 'false'.
1306 /sp:SamlToken/wsp:Policy/sp:RequireImplicitDerivedKeys
1307 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
1308 for this token to 'true' and the [Explicit Derived Keys] property for this token to
1309 'false'.
1310 /sp:SamlToken/wsp:Policy/sp:RequireKeyIdentifierReference
1311 This optional element indicates that a key identifier reference is required when
1312 referencing this token.
1313 /sp:SamlToken/wsp:Policy/sp:WssSamlV11Token10
1314 This optional element identifies that a SAML Version 1.1 token should be used as
1315 defined in [WSS: SAML Token Profile 1.0].
1316 /sp:SamlToken/wsp:Policy/sp:WssSamlV11Token11
1317 This optional element identifies that a SAML Version 1.1 token should be used as
1318 defined in [WSS: SAML Token Profile 1.1].
1319 /sp:SamlToken/wsp:Policy/sp:WssSamlV20Token11
1320 This optional element identifies that a SAML Version 2.0 token should be used as
1321 defined in [WSS: SAML Token Profile 1.1].
1322
1323 Note: This assertion does not describe how to obtain a SAML Token but rather assumes that
1324 both parties have the token already or have agreed separately on a mechanism for
1325 obtaining the token. If a definition of the mechanism for obtaining the SAML Token is
1326 desired in policy, the sp:IssuedToken assertion should be used instead.

1327 **5.3.9 RelToken Assertion**

1328 This element represents a requirement for a REL token.

1329 **Syntax**

```

1330 <sp:RelToken sp:IncludeToken="xs:anyURI"? ... >
1331   <wsp:Policy>
1332     (
1333       <sp:RequireDerivedKeys ... /> |
1334       <sp:RequireImplicitDerivedKeys ... /> |
1335       <sp:RequireExplicitDerivedKeys ... />
1336     ) ?
1337   <sp:RequireKeyIdentifierReference ... /> ?
1338   (
1339     <sp:WssRelV10Token10 ... /> |
1340     <sp:WssRelV20Token10 ... /> |
1341     <sp:WssRelV10Token11 ... /> |
1342     <sp:WssRelV20Token11 ... />
1343   ) ?
1344   ...
1345 </wsp:Policy> ?
1346   ...
1347 </sp:RelToken>

```

1348

1349 The following describes the attributes and elements listed in the schema outlined above:

1350 /sp:RelToken

1351 This identifies a RelToken assertion.

1352 /sp:RelToken/@sp:IncludeToken

1353 This optional attribute identifies the token inclusion value for this token assertion.

1354 /sp:RelToken/wsp:Policy

1355 This optional element identifies additional requirements for use of the sp:RelToken
1356 assertion.

1357 /sp:RelToken/wsp:Policy/sp:RequireDerivedKeys

1358 This optional element sets the [Derived Keys], [Explicit Derived Keys] and [Implicit
1359 Derived Keys] property for this token to 'true'.

1360 /sp:RelToken/wsp:Policy/sp:RequireExplicitDerivedKeys

1361 This optional element sets the [Derived Keys] and [Explicit Derived Keys] properties
1362 for this token to 'true' and the [Implicit Derived Keys] property for this token to
1363 'false'.

1364 /sp:RelToken/wsp:Policy/sp:RequireImplicitDerivedKeys

1365 This optional element sets the [Derived Keys] and [Implicit Derived Keys] properties
1366 for this token to 'true' and the [Explicit Derived Keys] property for this token to
1367 'false'.

1368 /sp:RelToken/wsp:Policy/sp:RequireKeyIdentifierReference

1369 This optional element indicates that a key identifier reference is required when
1370 referencing this token.

1371 /sp:RelToken/wsp:Policy/sp:WssRelV10Token10

1372 This optional element identifies that a REL Version 1.0 token should be used as
1373 defined in [WSS: REL Token Profile 1.0].

1374 /sp:RelToken/wsp:Policy/sp:WssRelV20Token10

1375 This optional element identifies that a REL Version 2.0 token should be used as
1376 defined in [WSS: REL Token Profile 1.0].

1377 /sp:RelToken/wsp:Policy/sp:WssRelV10Token11
1378 This optional element identifies that a REL Version 1.0 token should be used as
1379 defined in [WSS: REL Token Profile 1.1].

1380 /sp:RelToken/wsp:Policy/sp:WssRelV20Token11
1381 This optional element identifies that a REL Version 2.0 token should be used as
1382 defined in [WSS: REL Token Profile 1.1].

1383
1384 Note: This assertion does not describe how to obtain a REL Token but rather assumes that
1385 both parties have the token already or have agreed separately on a mechanism for
1386 obtaining the token. If a definition of the mechanism for obtaining the REL Token is desired
1387 in policy, the sp:IssuedToken assertion should be used instead.

1388 **5.3.10 HttpsToken Assertion**

1389 This element represents a requirement for a transport binding to support the use of HTTPS.

1390 **Syntax**

```
1391 <sp:HttpsToken ... >  
1392   <wsp:Policy>  
1393     (  
1394       <sp:HttpBasicAuthentication /> |  
1395       <sp:HttpDigestAuthentication /> |  
1396       <sp:RequireClientCertificate /> |  
1397       ...  
1398     )?  
1399     ...  
1400   </wsp:Policy> ?  
1401   ...  
1402 </sp:HttpsToken>
```

1403 The following describes the attributes and elements listed in the schema outlined above:

1404 /sp:HttpsToken

1405 This identifies an Https assertion stating that use of the HTTPS protocol specification
1406 is supported.

1407 /sp:HttpsToken/wsp:Policy

1408 This optional element identifies additional requirements for use of the sp:HttpsToken
1409 assertion.

1410 /sp:HttpsToken/wsp:Policy/sp:HttpBasicAuthentication

1411 This optional element indicates that the client MUST use HTTP Basic Authentication to
1412 authenticate to the service.

1413 /sp:HttpsToken/wsp:Policy/sp:HttpDigestAuthentication

1414 This optional element indicates that the client MUST use HTTP Digest Authentication
1415 to authenticate to the service.

1416 /sp:HttpsToken/wsp:Policy/sp:RequireClientCertificate

1417 This optional element indicates that the client MUST provide a certificate when
1418 negotiating the HTTPS session.

1419

1420

6 Security Binding Properties

1421 This section defines the various properties or conditions of a security binding, their
1422 semantics, values and defaults where appropriate. Properties are used by a binding in a
1423 manner similar to how variables are used in code. Assertions populate, (or set) the value of
1424 the property (or variable). When an assertion that populates a value of a property appears
1425 in a policy, that property is set to the value indicated by the assertion. The security binding
1426 then uses the value of the property to control its behavior. The properties listed here are
1427 common to the various security bindings described in Section 7. Assertions that define
1428 values for these properties are defined in Section 7. The following properties are used by
1429 the security binding assertions.

6.1 [Algorithm Suite] Property

1431 This property specifies the algorithm suite required for performing cryptographic operations
1432 with symmetric or asymmetric key based security tokens. An algorithm suite specifies actual
1433 algorithms and allowed key lengths. A policy alternative will define what algorithms are
1434 used and how they are used. This property defines the set of available algorithms. The
1435 value of this property is typically referenced by a security binding and is used to specify the
1436 algorithms used for all message level cryptographic operations performed under the security
1437 binding.

1438 Note: In some cases, this property MAY be referenced under a context other than a security
1439 binding and used to control the algorithms used under that context. For example,
1440 supporting token assertions define such a context. In such contexts, the specified
1441 algorithms still apply to message level cryptographic operations.

1442 An algorithm suite defines values for each of the following operations and properties:

1443	[Sym Sig]	Symmetric Key Signature
1444	[Asym Sig]	Signature with an asymmetric key
1445	[Dig]	Digest
1446	[Enc]	Encryption
1447	[Sym KW]	Symmetric Key Wrap
1448	[Asym KW]	Asymmetric Key Wrap
1449	[Comp Key]	Computed key
1450	[Enc KD]	Encryption key derivation
1451	[Sig KD]	Signature key derivation
1452	[Min SKL]	Minimum symmetric key length
1453	[Max SKL]	Maximum symmetric key length
1454	[Min AKL]	Minimum asymmetric key length
1455	[Max AKL]	Maximum asymmetric key length

1456

1457 The following table provides abbreviations for the algorithm URI used in the table below:

Abbreviation	Algorithm URI
HmacSha1	http://www.w3.org/2000/09/xmlldsig#hmac-sha1
RsaSha1	http://www.w3.org/2000/09/xmlldsig#rsa-sha1

Sha1	http://www.w3.org/2000/09/xmlldsig#sha1
Sha256	http://www.w3.org/2001/04/xmlenc#sha256
Sha512	http://www.w3.org/2001/04/xmlenc#sha512
Aes128	http://www.w3.org/2001/04/xmlenc#aes128-cbc
Aes192	http://www.w3.org/2001/04/xmlenc#aes192-cbc
Aes256	http://www.w3.org/2001/04/xmlenc#aes256-cbc
TripleDes	http://www.w3.org/2001/04/xmlenc#tripledes-cbc
KwAes128	http://www.w3.org/2001/04/xmlenc#kw-aes128
KwAes192	http://www.w3.org/2001/04/xmlenc#kw-aes192
KwAes256	http://www.w3.org/2001/04/xmlenc#kw-aes256
KwTripleDes	http://www.w3.org/2001/04/xmlenc#kw-tripledes
KwRsaOaep	http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p
KwRsa15	http://www.w3.org/2001/04/xmlenc#rsa-1_5
PSha1	http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/dk/p_sha1
PSha1L128	http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/dk/p_sha1
PSha1L192	http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/dk/p_sha1
PSha1L256	http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/dk/p_sha1
XPath	http://www.w3.org/TR/1999/REC-xpath-19991116
XPath20	http://www.w3.org/2002/06/xmlldsig-filter2
C14n	http://www.w3.org/2001/10/xml-c14n#
ExC14n	http://www.w3.org/2001/10/xml-exc-c14n#
SNT	http://www.w3.org/TR/soap12-n11n
STRT10	http://docs.oasis-open.org/wss/2004/xx/oasis-2004xx-wss-soap-message-security-1.0#STR-Transform
AbsXPath	http://docs.oasis-open.org/...TBD.../AbsXPath

1458

1459 The tables below show all the base algorithm suites defined by this specification. This table
 1460 defines values for properties which are common for all suites:

Property	Algorithm / Value
[Sym Sig]	HmacSha1
[Asym Sig]	RsaSha1
[Comp Key]	PSha1
[Max SKL]	256
[Min AKL]	1024
[Max AKL]	4096

1461 This table defines additional properties whose values can be specified along with the default
 1462 value for that property.

Property	Algorithm / Value
[C14n Algorithm]	ExC14n
[Soap Norm]	None
[STR Trans]	None
[XPath]	None

1463 This table defines values for the remaining components for each algorithm suite.

Algorithm Suite	[Dig]	[Enc]	[Sym KW]	[Asym KW]	[Enc KD]	[Sig KD]	[Min SKL]
Basic256	Sha1	Aes256	KwAes256	KwRsaOaep	PSha1L256	PSha1L192	256
Basic192	Sha1	Aes192	KwAes192	KwRsaOaep	PSha1L192	PSha1L192	192
Basic128	Sha1	Aes128	KwAes128	KwRsaOaep	PSha1L128	PSha1L128	128
TripleDes	Sha1	TripleDes	KwTripleDes	KwRsaOaep	PSha1L192	PSha1L192	192
Basic256Rsa15	Sha1	Aes256	KwAes256	KwRsa15	PSha1L256	PSha1L192	256
Basic192Rsa15	Sha1	Aes192	KwAes192	KwRsa15	PSha1L192	PSha1L192	192
Basic128Rsa15	Sha1	Aes128	KwAes128	KwRsa15	PSha1L128	PSha1L128	128
TripleDesRsa15	Sha1	TripleDes	KwTripleDes	KwRsa15	PSha1L192	PSha1L192	192
Basic256Sha256	Sha256	Aes256	KwAes256	KwRsaOaep	PSha1L256	PSha1L192	256
Basic192Sha256	Sha256	Aes192	KwAes192	KwRsaOaep	PSha1L192	PSha1L192	192
Basic128Sha256	Sha256	Aes128	KwAes128	KwRsaOaep	PSha1L128	PSha1L128	128
TripleDesSha256	Sha256	TripleDes	KwTripleDes	KwRsaOaep	PSha1L192	PSha1L192	192
Basic256Sha256Rsa15	Sha256	Aes256	KwAes256	KwRsa15	PSha1L256	PSha1L192	256
Basic192Sha256Rsa15	Sha256	Aes192	KwAes192	KwRsa15	PSha1L192	PSha1L192	192
Basic128Sha256Rsa15	Sha256	Aes128	KwAes128	KwRsa15	PSha1L128	PSha1L128	128
TripleDesSha256Rsa15	Sha256	TripleDes	KwTripleDes	KwRsa15	PSha1L192	PSha1L192	192

1464 **6.2 [Timestamp] Property**

1465 This boolean property specifies whether a `wsu:Timestamp` element is present in the
1466 `wsse:Security` header. If the value is 'true', the timestamp element MUST be present and
1467 MUST be integrity protected either by transport or message level security. If the value is
1468 'false', the timestamp element MUST NOT be present. The default value for this property is
1469 'false'.

1470 **6.3 [Protection Order] Property**

1471 This property indicates the order in which integrity and confidentiality are applied to the
1472 message, in cases where both integrity and confidentiality are required:

EncryptBeforeSigning	Signature MUST be computed over ciphertext. Encryption key and signing key MUST be derived from the same source key.
SignBeforeEncrypting	Signature MUST be computed over plaintext. The resulting signature SHOULD be encrypted. Supporting signatures MUST be over the plain text signature.

1473 The default value for this property is 'SignBeforeEncrypting'.

1474 **6.4 [Signature Protection] Property**

1475 This boolean property specifies whether the signature must be encrypted. If the value is
1476 'true', the primary signature MUST be encrypted and any signature confirmation elements
1477 MUST also be encrypted. If the value is 'false', the primary signature MUST NOT be
1478 encrypted and any signature confirmation elements MUST NOT be encrypted. The default
1479 value for this property is 'false'.

1480 **6.5 [Token Protection] Property**

1481 This boolean property specifies whether signatures must cover the token used to generate
1482 that signature. If the value is 'true', then each token used to generate a signature MUST be
1483 covered by that signature. If the value is 'false', then the token MUST NOT be covered by
1484 the signature. Note that in cases where derived keys are used, the 'main' token and NOT
1485 the derived key token is covered by the signature. It is recommended that assertions that
1486 define values for this property apply to [Endpoint Policy Subject]. The default value for this
1487 property is 'false'.

1488 **6.6 [Entire Header and Body Signatures] Property**

1489 This boolean property specifies whether signature digests over the SOAP body and SOAP
1490 headers must only cover the entire body and entire header elements. If the value is 'true',
1491 then each digest over the SOAP body MUST be over the entire SOAP body element and not
1492 a descendant of that element. In addition each digest over a SOAP header MUST be over an
1493 actual header element and not a descendant of a header element. This restriction does not
1494 specifically apply to the wsse:Security header. However signature digests over child
1495 elements of the wsse:Security header MUST be over the entire child element and not a
1496 descendent of that element. If the value is 'false', then signature digests MAY be over a
1497 descendant of the SOAP Body or a descendant of a header element. Setting the value of this
1498 property to 'true' mitigates against some possible re-writing attacks. It is recommended
1499 that assertions that define values for this property apply to [Endpoint Policy Subject]. The
1500 default value for this property is 'false'.

1501 **6.7 [Security Header Layout] Property**

1502 This property indicates which layout rules to apply when adding items to the security
1503 header. The following table shows which rules are defined by this specification.

Strict	Items are added to the security header following the numbered layout rules described below according to a general principle of 'declare before use'.
Lax	Items are added to the security header in any order that conforms to WSS: SOAP Message Security
LaxTimestampFirst	As Lax except that the first item in the security header MUST be a wsse:Timestamp. Note that the [Timestamp] property MUST also be set to 'true' in this case.
LaxTimestampLast	As Lax except that the last item in the security header MUST be a wsse:Timestamp. Note that the [Timestamp] property MUST also be set to 'true' in this case.

1504

1505 6.7.1 Strict Layout Rules

- 1506 1. Tokens that are included in the message MUST be declared before use. For example,
 - 1507 a. A local signing token MUST occur before the signature that uses it.
 - 1508 b. A local token serving as the source token for a derived key token MUST occur
 - 1509 before that derived key token.
 - 1510 c. A local encryption token MUST occur before the reference list that points to
 - 1511 `xenc:EncryptedData` elements that use it.
 - 1512 d. If the same token is used for both signing and encryption, then it should
 - 1513 appear before the earlier element in the security header.
- 1514 2. Signed elements inside the security header MUST occur before the signature that
- 1515 signs them. For example,
 - 1516 a. A timestamp MUST occur before the signature that signs it.
 - 1517 b. A Username token (usually in encrypted form) MUST occur before the
 - 1518 signature that signs it.
 - 1519 c. A primary signature MUST occur before the supporting token signature that
 - 1520 signs the primary signature's signature value element.
 - 1521 d. A `wsse11:SignatureConfirmation` element MUST occur before the signature
 - 1522 that signs it.
- 1523 3. When an element in a security header is encrypted, the resulting
- 1524 `xenc:EncryptedData` element has the same order requirements as the source plain
- 1525 text element. For example, an encrypted primary signature MUST occur before any
- 1526 supporting token signature per 2c above and an encrypted token has the same
- 1527 ordering requirements as the unencrypted token.
- 1528 4. If there are any encrypted elements in the message then a top level
- 1529 `xenc:ReferenceList` element MUST be present in the security header. The
- 1530 `xenc:ReferenceList` MUST occur before any `xenc:EncryptedData` elements in the
- 1531 security header that are referenced from the reference list. However, the
- 1532 `xenc:ReferenceList` is not required to appear before independently encrypted tokens
- 1533 such as the `xenc:EncryptedKey` token as defined in WSS.
- 1534 5. An `xenc:EncryptedKey` element without an internal reference list [[WSS: SOAP](#)
- 1535 [Message Security 1.1](#)] MUST obey rule (1). An `xenc:EncryptedKey` element with an
- 1536 internal reference list MUST additionally obey rule (4).

1537 Examples of these layout rules for each security binding are described in Appendix C.

1538

7 Security Binding Assertions

1539

The appropriate representation of the different facets of security mechanisms requires

1540

distilling the common primitives (to enable reuse) and then combining the primitive

1541

elements into patterns. The policy scope of assertions defined in this section is the policy

1542

scope of their containing element.

1543

7.1 AlgorithmSuite Assertion

1544

This assertion indicates a requirement for an algorithm suite as defined under the

1545

[Algorithm Suite] property described in Section 6.1. The scope of this assertion is defined by

1546

its containing assertion.

1547

Syntax

1548

```
<sp:AlgorithmSuite ... >
  <wsp:Policy>
    (<sp:Basic256 ... /> |
     <sp:Basic192 ... /> |
     <sp:Basic128 ... /> |
     <sp:TripleDes ... /> |
     <sp:Basic256Rsa15 ... /> |
     <sp:Basic192Rsa15 ... /> |
     <sp:Basic128Rsa15 ... /> |
     <sp:TripleDesRsa15 ... /> |
     <sp:Basic256Sha256 ... /> |
     <sp:Basic192Sha256 ... /> |
     <sp:Basic128Sha256 ... /> |
     <sp:TripleDesSha256 ... /> |
     <sp:Basic256Sha256Rsa15 ... /> |
     <sp:Basic192Sha256Rsa15 ... /> |
     <sp:Basic128Sha256Rsa15 ... /> |
     <sp:TripleDesSha256Rsa15 ... /> |
     ...)
    <sp:InclusiveC14N ... /> ?
    <sp:SOAPNormalization10 ... /> ?
    <sp:STRTransform10 ... /> ?
    (<sp:XPath10 ... /> |
     <sp:XPathFilter20 ... /> |
     <sp:AbsXPath ... /> |
     ...)?
    ...
  </wsp:Policy>
  ...
</sp:AlgorithmSuite>
```

1578

The following describes the attributes and elements listed in the schema outlined above:

1579

/sp:AlgorithmSuite

1580

This identifies an AlgorithmSuite assertion.

1581

/sp:AlgorithmSuite/wsp:Policy

1582

This element contains one or more policy assertions that indicate the specific algorithm suite to use.

1583

1584

/sp:AlgorithmSuite/wsp:Policy/sp:Basic256

1585

This assertion indicates that the [Algorithm Suite] property is set to 'Basic256'.

1586

1587 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192
1588 This assertion indicates that the [Algorithm Suite] property is set to 'Basic192'.
1589 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128
1590 This assertion indicates that the [Algorithm Suite] property is set to 'Basic128'.
1591 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDes
1592 This assertion indicates that the [Algorithm Suite] property is set to 'TripleDes'.
1593 /sp:AlgorithmSuite/wsp:Policy/sp:Basic256Rsa15
1594 This assertion indicates that the [Algorithm Suite] property is set to 'Basic256Rsa15'.
1595 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192Rsa15
1596 This assertion indicates that the [Algorithm Suite] property is set to 'Basic192Rsa15'.
1597 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128Rsa15
1598 This assertion indicates that the [Algorithm Suite] property is set to 'Basic128Rsa15'.
1599 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDesRsa15
1600 This assertion indicates that the [Algorithm Suite] property is set to
1601 'TripleDesRsa15'.
1602 /sp:AlgorithmSuite/wsp:Policy/sp:Basic256Sha256
1603 This assertion indicates that the [Algorithm Suite] property is set to
1604 'Basic256Sha256'.
1605 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192Sha256
1606 This assertion indicates that the [Algorithm Suite] property is set to
1607 'Basic192Sha256'.
1608 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128Sha256
1609 This assertion indicates that the [Algorithm Suite] property is set to
1610 'Basic128Sha256'.
1611 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDesSha256
1612 This assertion indicates that the [Algorithm Suite] property is set to
1613 'TripleDesSha256'.
1614 /sp:AlgorithmSuite/wsp:Policy/sp:Basic256Sha256Rsa15
1615 This assertion indicates that the [Algorithm Suite] property is set to
1616 'Basic256Sha256Rsa15'.
1617 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192Sha256Rsa15
1618 This assertion indicates that the [Algorithm Suite] property is set to
1619 'Basic192Sha256Rsa15'.
1620 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128Sha256Rsa15
1621 This assertion indicates that the [Algorithm Suite] property is set to
1622 'Basic128Sha256Rsa15'.
1623 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDesSha256Rsa15
1624 This assertion indicates that the [Algorithm Suite] property is set to
1625 'TripleDesSha256Rsa15'.
1626 /sp:AlgorithmSuite/wsp:Policy/sp:InclusiveC14N

1627 This assertion indicates that the [C14N] property of an algorithm suite is set to
1628 'C14N'. Note: as indicated in Section 6.1 the default value of the [C14N] property is
1629 'ExcC14N'.

1630 /sp:AlgorithmSuite/wsp:Policy/sp:SoapNormalization10

1631 This assertion indicates that the [SOAP Norm] property is set to 'SNT'.

1632 /sp:AlgorithmSuite/wsp:Policy/sp:STRTransform10

1633 This assertion indicates that the [STR Transform] property is set to 'STRT10'.

1634 /sp:AlgorithmSuite/wsp:Policy/sp:XPath10

1635 This assertion indicates that the [XPath] property is set to 'XPath'.

1636 /sp:AlgorithmSuite/wsp:Policy/sp:XPathFilter20

1637 This assertion indicates that the [XPath] property is set to 'XPath20'.

1638 /sp:AlgorithmSuite/wsp:Policy/sp:AbsXPath

1639 This assertion indicates that the [XPath] property is set to 'AbsXPath' (see
1640 [AbsoluteLocationPath](#) in [XPATH]).

1641

1642 7.2 Layout Assertion

1643 This assertion indicates a requirement for a particular security header layout as defined
1644 under the [Security Header Layout] property described in Section 6.7. The scope of this
1645 assertion is defined by its containing assertion.

1646 Syntax

```
1647 <sp:Layout ... >  
1648   <wsp:Policy>  
1649     <sp:Strict ... /> |  
1650     <sp:Lax ... /> |  
1651     <sp:LaxTsFirst ... /> |  
1652     <sp:LaxTsLast ... /> |  
1653     ...  
1654   </wsp:Policy>  
1655   ...  
1656 </sp:Layout>
```

1657

1658 The following describes the attributes and elements listed in the schema outlined above:

1659 /sp:Layout

1660 This identifies a Layout assertion.

1661 /sp:Layout/wsp:Policy

1662 This element contains one or more policy assertions that indicate the specific security
1663 header layout to use.

1664 /sp:Layout/wsp:Policy/sp:Strict

1665 This assertion indicates that the [Security Header Layout] property is set to 'Strict'.

1666 /sp:Layout/wsp:Policy/sp:Lax

1667 This assertion indicates that the [Security Header Layout] property is set to 'Lax'.

1668 /sp:Layout/wsp:Policy/sp:LaxTsFirst

1669 This assertion indicates that the [Security Header Layout] property is set to
1670 'LaxTimestampFirst'. Note that the [Timestamp] property MUST also be set to 'true'
1671 by the presence of an sp:IncludeTimestamp assertion.

1672 /sp:Layout/wsp:Policy/sp:LaxTsLast

1673 This assertion indicates that the [Security Header Layout] property is set to
1674 'LaxTimestampLast'. Note that the [Timestamp] property MUST also be set to 'true'
1675 by the presence of an sp:IncludeTimestamp assertion.

1676 7.3 TransportBinding Assertion

1677 The TransportBinding assertion is used in scenarios in which message protection and
1678 security correlation is provided by means other than [WSS: SOAP Message Security](#), for
1679 example by a secure transport like HTTPS. Specifically, this assertion indicates that the
1680 message is protected using the means provided by the transport. This binding has one
1681 binding specific token property; [Transport Token]. This assertion MUST apply to [Endpoint
1682 Policy Subject].

1683 Syntax

```
1684 <sp:TransportBinding ... >  
1685   <wsp:Policy>  
1686     <sp:TransportToken ... >  
1687       <wsp:Policy> ... </wsp:Policy>  
1688       ...  
1689     </sp:TransportToken>  
1690     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite>  
1691     <sp:Layout ... > ... </sp:Layout> ?  
1692     <sp:IncludeTimestamp ... /> ?  
1693     ...  
1694   </wsp:Policy>  
1695   ...  
1696 </sp:TransportBinding>
```

1697
1698 The following describes the attributes and elements listed in the schema outlined above:

1699 /sp:TransportBinding

1700 This identifies a TransportBinding assertion.

1701 /sp:TransportBinding/wsp:Policy

1702 This indicates a nested `wsp:Policy` element that defines the behavior of the
1703 TransportBinding assertion.

1704 /sp:TransportBinding/wsp:Policy/sp:TransportToken

1705 This assertion indicates a requirement for a Transport Token. The specified token
1706 populates the [Transport Token] property and indicates how the transport is secured.

1707 /sp:TransportBinding/wsp:Policy/sp:TransportToken/wsp:Policy

1708 This indicates a nested policy that identifies the type of Transport Token to use.

1709 /sp:TransportBinding/wsp:Policy/sp:AlgorithmSuite

1710 This assertion indicates a value that populates the [Algorithm Suite] property. See
1711 Section 6.1 for more details.

1712 /sp:TransportBinding/wsp:Policy/sp:Layout

1713 This assertion indicates a value that populates the [Security Header Layout]
1714 property. See Section 6.7 for more details.

1715 /sp:TransportBinding/wsp:Policy/sp:IncludeTimestamp

1716 This assertion indicates that the [Timestamp] property is set to 'true'.

1717 7.4 SymmetricBinding Assertion

1718 The SymmetricBinding assertion is used in scenarios in which message protection is
1719 provided by means defined in [WSS: SOAP Message Security](#). This binding has two binding
1720 specific token properties; [Encryption Token] and [Signature Token]. If the message pattern
1721 requires multiple messages, this binding defines that the [Encryption Token] used from
1722 initiator to recipient is also used from recipient to initiator. Similarly, the [Signature Token]
1723 used from initiator to recipient is also use from recipient to initiator. If a sp:ProtectionToken
1724 assertion is specified, the specified token populates both token properties and is used as the
1725 basis for both encryption and signature in both directions. This assertion SHOULD apply to
1726 [Endpoint Policy Subject]. This assertion MAY apply to [Operation Policy Subject].

1727 Syntax

```
1728 <sp:SymmetricBinding ... >  
1729   <wsp:Policy>  
1730     (  
1731       <sp:EncryptionToken ... >  
1732         <wsp:Policy> ... </wsp:Policy>  
1733       </sp:EncryptionToken>  
1734       <sp:SignatureToken ... >  
1735         <wsp:Policy> ... </wsp:Policy>  
1736       </sp:SignatureToken>  
1737     ) | (  
1738       <sp:ProtectionToken ... >  
1739         <wsp:Policy> ... </wsp:Policy>  
1740       </sp:ProtectionToken>  
1741     )  
1742   <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite>  
1743   <sp:Layout ... > ... </sp:Layout> ?  
1744   <sp:IncludeTimestamp ... /> ?  
1745   <sp:EncryptBeforeSigning ... /> ?  
1746   <sp:EncryptSignature ... /> ?  
1747   <sp:ProtectTokens ... /> ?  
1748   <sp:OnlySignEntireHeadersAndBody ... /> ?  
1749   ...  
1750 </wsp:Policy>  
1751   ...  
1752 </sp:SymmetricBinding>
```

1753

1754 The following describes the attributes and elements listed in the schema outlined above:

1755 /sp:SymmetricBinding

1756 This identifies a SymmetricBinding assertion.

1757 /sp:SymmetricBinding/wsp:Policy

1758 This indicates a nested `wsp:Policy` element that defines the behavior of the
1759 SymmetricBinding assertion.

1760 /sp:SymmetricBinding/wsp:Policy/sp:EncryptionToken

1761 This assertion indicates a requirement for an Encryption Token. The specified token
1762 populates the [Encryption Token] property and is used for encryption. It is an error
1763 for both an `sp:EncryptionToken` and an `sp:ProtectionToken` assertion to be specified.

1764 /sp:SymmetricBinding/wsp:Policy/sp:EncryptionToken/wsp:Policy

1765 The policy contained here MUST identify exactly one token to use for encryption.

1766 /sp:SymmetricBinding/wsp:Policy/sp:SignatureToken

1767 This assertion indicates a requirement for a Signature Token. The specified token
1768 populates the [Signature Token] property and is used for the message signature. It
1769 is an error for both an sp:SignatureToken and an sp:ProtectionToken assertion to be
1770 specified.

1771 /sp:SymmetricBinding/wsp:Policy/sp:SignatureToken/wsp:Policy

1772 The policy contained here MUST identify exactly one token to use for signatures.

1773 /sp:SymmetricBinding/wsp:Policy/sp:ProtectionToken

1774 This assertion indicates a requirement for a Protection Token. The specified token
1775 populates the [Encryption Token] and [Signature Token properties] and is used for
1776 the message signature and for encryption. It is an error for both an
1777 sp:ProtectionToken assertion and either an sp:EncryptionToken assertion or an
1778 sp:SignatureToken assertion to be specified.

1779 /sp:SymmetricBinding/wsp:Policy/sp:ProtectionToken/wsp:Policy

1780 The policy contained here MUST identify exactly one token to use for protection.

1781 /sp:SymmetricBinding/wsp:Policy/sp:AlgorithmSuite

1782 This assertion indicates a value that populates the [Algorithm Suite] property. See
1783 Section 6.1 for more details.

1784 /sp:SymmetricBinding/wsp:Policy/sp:Layout

1785 This assertion indicates a value that populates the [Security Header Layout]
1786 property. See Section 6.7 for more details.

1787 /sp:SymmetricBinding/wsp:Policy/sp:IncludeTimestamp

1788 This assertion indicates that the [Timestamp] property is set to 'true'.

1789 /sp:SymmetricBinding/wsp:Policy/sp:EncryptBeforeSigning

1790 This assertion indicates that the [Protection Order] property is set to
1791 'EncryptBeforeSigning'.

1792 /sp:SymmetricBinding/wsp:Policy/sp:EncryptSignature

1793 This assertion indicates that the [Signature Protection] property is set to 'true'.

1794 /sp:SymmetricBinding/wsp:Policy/sp:ProtectTokens

1795 This assertion indicates that the [Token Protection] property is set to 'true'.

1796 /sp:SymmetricBinding/wsp:Policy/sp:OnlySignEntireHeadersAndBody

1797 This assertion indicates that the [Entire Header And Body Signatures] property is set
1798 to 'true'.

1799 **7.5 AsymmetricBinding Assertion**

1800 The AsymmetricBinding assertion is used in scenarios in which message protection is
1801 provided by means defined in WSS: SOAP Message Security using asymmetric key (Public
1802 Key) technology. Commonly used asymmetric algorithms, such as RSA, allow the same key
1803 pair to be used for both encryption and signature. However it is also common practice to
1804 use distinct keys for encryption and signature, because of their different lifecycles.
1805

1806 This binding enables either of these practices by means of four binding specific token
1807 properties: [Initiator Signature Token], [Initiator Encryption Token], [Recipient Signature
1808 Token] and [Recipient Encryption Token].

1809
1810 If the same key pair is used for signature and encryption, then [Initiator Signature Token]
1811 and [Initiator Encryption Token] will both refer to the same token. Likewise [Recipient
1812 Signature Token] and [Recipient Encryption Token] will both refer to the same token.

1813
1814 If distinct key pairs are used for signature and encryption, then [Initiator Signature Token]
1815 and [Initiator Encryption Token] will refer to different tokens. Likewise [Recipient Signature
1816 Token] and [Recipient Encryption Token] will refer to different tokens.

1817
1818 If the message pattern requires multiple messages, the [Initiator Signature Token] is used
1819 for the message signature from initiator to the recipient. The [Initiator Encryption Token] is
1820 used for the response message encryption from recipient to the initiator. The [Recipient
1821 Signature Token] is used for the response message signature from recipient to the initiator.
1822 The [Recipient Encryption Token] is used for the message encryption from initiator to the
1823 recipient. Note that in each case, the token is associated with the party (initiator or
1824 recipient) who knows the secret.

1825 This assertion SHOULD apply to [Endpoint Policy Subject]. This assertion MAY apply to
1826 [Operation Policy Subject].

1827 **Syntax**

```
1828 <sp:AsymmetricBinding ... >  
1829   <wsp:Policy>  
1830     <sp:InitiatorToken>  
1831       <wsp:Policy> ... </wsp:Policy>  
1832     </sp:InitiatorToken>  
1833     <sp:RecipientToken>  
1834       <wsp:Policy> ... </wsp:Policy>  
1835     </sp:RecipientToken>  
1836     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite>  
1837     <sp:Layout ... > ... </sp:Layout> ?  
1838     <sp:IncludeTimestamp ... /> ?  
1839     <sp:EncryptBeforeSigning ... /> ?  
1840     <sp:EncryptSignature ... /> ?  
1841     <sp:ProtectTokens ... /> ?  
1842     <sp:OnlySignEntireHeadersAndBody ... /> ?  
1843     ...  
1844   </wsp:Policy>  
1845   ...  
1846 </sp:AsymmetricBinding>
```

1847
1848 The following describes the attributes and elements listed in the schema outlined above:

1849 /sp:AsymmetricBinding

1850 This identifies a AsymmetricBinding assertion.

1851 /sp:AsymmetricBinding/wsp:Policy

1852 This indicates a nested `wsp:Policy` element that defines the behavior of the
1853 AsymmetricBinding assertion.

1854 /sp:AsymmetricBinding/wsp:Policy/sp:InitiatorToken

1855 This assertion indicates a requirement for an Initiator Token. The specified token
1856 populates the [Initiator Signature Token] and [Initiator Encryption Token] properties
1857 and is used for the message signature from initiator to recipient, and encryption from
1858 recipient to initiator.

1859 /sp:AsymmetricBinding/wsp:Policy/sp:InitiatorToken/wsp:Policy

1860 The policy contained here MUST identify one or more token assertions.

1861 /sp:AsymmetricBinding/wsp:Policy/sp:RecipientToken

1862 This assertion indicates a requirement for a Recipient Token. The specified token
1863 populates the [Recipient Signature Token] and [Recipient Encryption Token] property
1864 and is used for encryption from initiator to recipient, and for the message signature
1865 from recipient to initiator.

1866 /sp:AsymmetricBinding/wsp:Policy/sp:RecipientToken/wsp:Policy

1867 The policy contained here MUST identify one or more token assertions.

1868 /sp:AsymmetricBinding/wsp:Policy/sp:InitiatorSignatureToken

1869 This assertion indicates a requirement for an Initiator Signature Token. The specified
1870 token populates the [Initiator Signature Token] property and is used for the message
1871 signature from initiator to recipient.

1872 /sp:AsymmetricBinding/wsp:Policy/sp:InitiatorSignatureToken/wsp:Policy

1873 The policy contained here MUST identify one or more token assertions.

1874 /sp:AsymmetricBinding/wsp:Policy/sp:InitiatorEncryptionToken

1875 This assertion indicates a requirement for an Initiator Encryption Token. The
1876 specified token populates the [Initiator Encryption Token] property and is used for
1877 the message encryption from recipient to initiator.

1878 /sp:AsymmetricBinding/wsp:Policy/sp:InitiatorEncryptionToken/wsp:Policy

1879 The policy contained here MUST identify one or more token assertions.

1880 /sp:AsymmetricBinding/wsp:Policy/sp:RecipientSignatureToken

1881 This assertion indicates a requirement for an Recipient Signature Token. The
1882 specified token populates the [Recipient Signature Token] property and is used for
1883 the message signature from Recipient to recipient.

1884 /sp:AsymmetricBinding/wsp:Policy/sp:RecipientSignatureToken/wsp:Policy

1885 The policy contained here MUST identify one or more token assertions.

1886 /sp:AsymmetricBinding/wsp:Policy/sp:RecipientEncryptionToken

1887 This assertion indicates a requirement for an Recipient Encryption Token. The
1888 specified token populates the [Recipient Encryption Token] property and is used for
1889 the message encryption from recipient to Recipient.

1890 /sp:AsymmetricBinding/wsp:Policy/sp:RecipientEncryptionToken/wsp:Policy

1891 The policy contained here MUST identify one or more token assertions.

1892 /sp:AsymmetricBinding/wsp:Policy/sp:AlgorithmSuite

1893 This assertion indicates a value that populates the [Algorithm Suite] property. See
1894 Section 6.1 for more details.

1895 /sp:AsymmetricBinding/wsp:Policy/sp:Layout

1896 This assertion indicates a value that populates the [Security Header Layout]
1897 property. See Section 6.7 for more details.

1898 /sp:AsymmetricBinding/wsp:Policy/sp:IncludeTimestamp
1899 This assertion indicates that the [Timestamp] property is set to 'true'.

1900 /sp:AsymmetricBinding/wsp:Policy/sp:EncryptBeforeSigning
1901 This assertion indicates that the [Protection Order] property is set to
1902 'EncryptBeforeSigning'.

1903 /sp:AsymmetricBinding/wsp:Policy/sp:EncryptSignature
1904 This assertion indicates that the [Signature Protection] property is set to 'true'.

1905 /sp:AsymmetricBinding/wsp:Policy/sp:ProtectTokens
1906 This assertion indicates that the [Token Protection] property is set to 'true'.

1907 /sp:AsymmetricBinding/wsp:Policy/sp:OnlySignEntireHeadersAndBody
1908 This assertion indicates that the [Entire Header And Body Signatures] property is set
1909 to 'true'.

1910

8 Supporting Tokens

1911 Security Bindings use tokens to secure the message exchange. The Security Binding will
1912 require one to create a signature using the token identified in the Security Binding policy.
1913 This signature will here-to-fore be referred to as the "message signature". Additional tokens
1914 may be specified to augment the claims provided by the token associated with the
1915 "message signature" provided by the Security Binding. This section defines four properties
1916 related to supporting token requirements which may be referenced by a Security Binding:
1917 [Supporting Tokens], [Signed Supporting Tokens], [Endorsing Supporting Tokens] and
1918 [Signed Endorsing Supporting Tokens]. Four assertions are defined to populate those
1919 properties: SupportingTokens, SignedSupportingTokens, EndorsingSupportingTokens, and
1920 SignedEndorsingSupportingTokens. These assertions SHOULD apply to [Endpoint Policy
1921 Subject]. These assertions MAY apply to [Message Policy Subject] or [Operation Policy
1922 Subject].

1923

1924 Supporting tokens may be specified at a different scope than the binding assertion which
1925 provides support for securing the exchange. For instance, a binding is specified at the scope
1926 of an endpoint, while the supporting tokens might be defined at the scope of a message.
1927 When assertions that populate this property are defined in overlapping scopes, the sender
1928 should merge the requirements by including all tokens from the outer scope and any
1929 additional tokens for a specific message from the inner scope.

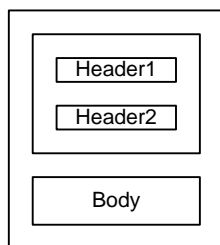
1930

1931 In cases where multiple tokens are specified that sign and/or encrypt overlapping message
1932 parts, all the tokens should sign and encrypt the various message parts. In such cases
1933 ordering of elements (tokens, signatures, reference lists etc.) in the security header would
1934 be used to determine which order signature and encryptions occurred in.

1935

1936 To illustrate the different ways that supporting tokens may be bound to the message, let's
1937 consider a message with three components: Header1, Header2, and Body.

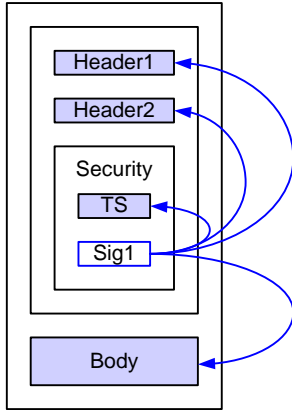
1938



1939

1940 Even before any supporting tokens are added, each binding requires that the message is
1941 signed using a token satisfying the required usage for that binding, and that the signature
1942 (Sig1) covers important parts of the message including the message timestamp (TS)
1943 facilitate replay detection. The signature is then included as part of the Security header as
1944 illustrated below:

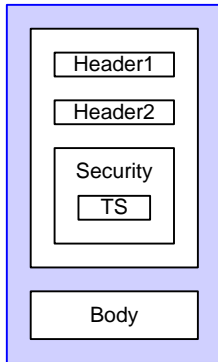
1945



1946

1947 Note: if required, the initiator may also include in the Security header the token used as the
 1948 basis for the message signature (Sig1), not shown in the diagram.

1949 If transport security is used, only the message timestamp (TS) is included in the Security
 1950 header as illustrated below:



1951

1952 8.1 Supporting Tokens Assertion

1953 Supporting tokens are included in the security header and may optionally include additional
 1954 message parts to sign and/or encrypt.

1955 Syntax

```

1956 <sp:SupportingTokens ... >
1957   <wsp:Policy>
1958     [Token Assertion]+
1959     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite> ?
1960     (
1961       <sp:SignedParts ... > ... </sp:SignedParts> |
1962       <sp:SignedElements ... > ... </sp:SignedElements> |
1963       <sp:EncryptedParts ... > ... </sp:EncryptedParts> |
1964       <sp:EncryptedElements ... > ... </sp:EncryptedElements> |
1965     ) *
1966     ...
1967   </wsp:Policy>
1968   ...
1969 </sp:SupportingTokens>
  
```

1970

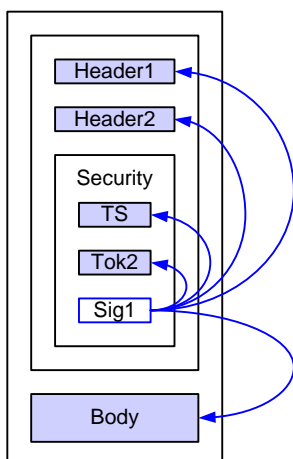
1971 The following describes the attributes and elements listed in the schema outlined above:

1972 /sp:SupportingTokens

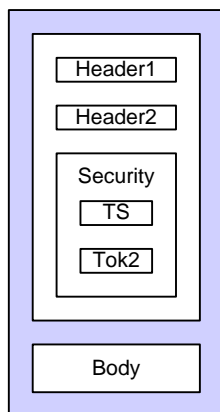
1973 This identifies a SupportingTokens assertion. The specified tokens populate the
 1974 [Supporting Tokens] property.
 1975 /sp:SupportingTokens/wsp:Policy
 1976 This describes additional requirements for satisfying the SupportingTokens assertion.
 1977 /sp:SupportingTokens/wsp:Policy/[Token Assertion]
 1978 The policy MUST identify one or more token assertions.
 1979 /sp:SupportingTokens/wsp:Policy/sp:AlgorithmSuite
 1980 This optional element follows the schema outlined in Section 7.1 and describes the
 1981 algorithms to use for cryptographic operations performed with the tokens identified
 1982 by this policy assertion.
 1983 /sp:SupportingTokens/wsp:Policy/sp:SignedParts
 1984 This optional element follows the schema outlined in Section 4.1.1 and describes
 1985 additional message parts that MUST be included in the signature generated with the
 1986 token identified by this policy assertion.
 1987 /sp:SupportingTokens/wsp:Policy/sp:SignedElements
 1988 This optional element follows the schema outlined in Section 4.1.2 and describes
 1989 additional message elements that MUST be included in the signature generated with
 1990 the token identified by this policy assertion.
 1991 /sp:SupportingTokens/wsp:Policy/sp:EncryptedParts
 1992 This optional element follows the schema outlined in Section 4.2.1 and describes
 1993 additional message parts that MUST be encrypted using the token identified by this
 1994 policy assertion.
 1995 /sp:SupportingTokens/wsp:Policy/sp:EncryptedElements
 1996 This optional element follows the schema outlined in Section 4.2.2 and describes
 1997 additional message elements that MUST be encrypted using the token identified by
 1998 this policy assertion.

1999 **8.2 SignedSupportingTokens Assertion**

2000 Signed tokens are included in the "message signature" as defined above and may optionally
 2001 include additional message parts to sign and/or encrypt. The diagram below illustrates how
 2002 the attached token (Tok2) is signed by the message signature (Sig1):
 2003



2005 If transport security is used, the token (Tok2) is included in the Security header as
2006 illustrated below:
2007



2008

2009 Syntax

2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023

```
<sp:SignedSupportingTokens ... >  
  <wsp:Policy>  
    [Token Assertion]+  
    <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite> ?  
    (  
      <sp:SignedParts ... > ... </sp:SignedParts> |  
      <sp:SignedElements ... > ... </sp:SignedElements> |  
      <sp:EncryptedParts ... > ... </sp:EncryptedParts> |  
      <sp:EncryptedElements ... > ... </sp:EncryptedElements> |  
    ) *  
    ...  
  </wsp:Policy>  
  ...  
</sp:SignedSupportingTokens>
```

2024

2025 The following describes the attributes and elements listed in the schema outlined above:

2026 /sp:SignedSupportingTokens

2027 This identifies a SignedSupportingTokens assertion. The specified tokens populate the
2028 [Signed Supporting Tokens] property.

2029 /sp:SignedSupportingTokens/wsp:Policy

2030 This describes additional requirements for satisfying the SignedSupportingTokens
2031 assertion.

2032 /sp:SignedSupportingTokens/wsp:Policy/[Token Assertion]

2033 The policy MUST identify one or more token assertions.

2034 /sp:SignedSupportingTokens/wsp:Policy/sp:AlgorithmSuite

2035 This optional element follows the schema outlined in Section 7.1 and describes the
2036 algorithms to use for cryptographic operations performed with the tokens identified
2037 by this policy assertion.

2038 /sp:SignedSupportingTokens/wsp:Policy/sp:SignedParts

2039 This optional element follows the schema outlined in Section 4.1.1 and describes
2040 additional message parts that MUST be included in the signature generated with the
2041 token identified by this policy assertion.

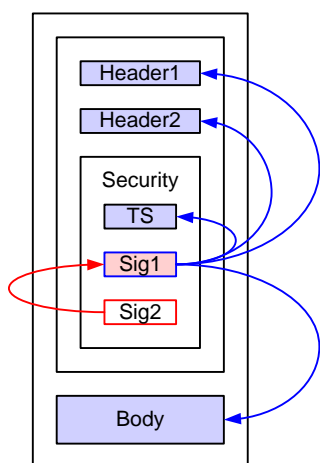
2042 /sp:SignedSupportingTokens/wsp:Policy/sp:SignedElements
2043 This optional element follows the schema outlined in Section 4.1.2 and describes
2044 additional message elements that MUST be included in the signature generated with
2045 the token identified by this policy assertion.

2046 /sp:SignedSupportingTokens/wsp:Policy/sp:EncryptedParts
2047 This optional element follows the schema outlined in Section 4.2.1 and describes
2048 additional message parts that MUST be encrypted using the token identified by this
2049 policy assertion.

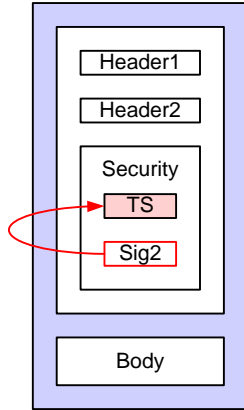
2050 /sp:SignedSupportingTokens/wsp:Policy/sp:EncryptedElements
2051 This optional element follows the schema outlined in Section 4.2.2 and describes
2052 additional message elements that MUST be encrypted using the token identified by
2053 this policy assertion.

2054 **8.3 EndorsingSupportingTokens Assertion**

2055 Endorsing tokens sign the message signature, that is they sign the entire `ds:Signature`
2056 element produced from the message signature and may optionally include additional
2057 message parts to sign and/or encrypt. The diagram below illustrates how the endorsing
2058 signature (Sig2) signs the message signature (Sig1):
2059



2060
2061 If transport security is used, the signature (Sig2) should cover the message timestamp as
2062 illustrated below:
2063



2064

2065 **Syntax**

```

2066 <sp:EndorsingSupportingTokens ... >
2067   <wsp:Policy>
2068     [Token Assertion]+
2069     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite ?
2070     (
2071       <sp:SignedParts ... > ... </sp:SignedParts> |
2072       <sp:SignedElements ... > ... </sp:SignedElements> |
2073       <sp:EncryptedParts ... > ... </sp:EncryptedParts> |
2074       <sp:EncryptedElements ... > ... </sp:EncryptedElements> |
2075     ) *
2076     ...
2077   </wsp:Policy>
2078   ...
2079 </sp:EndorsingSupportingTokens>

```

2080

2081 The following describes the attributes and elements listed in the schema outlined above:

2082 /sp:EndorsingSupportingTokens

2083 This identifies an EndorsingSupportingTokens assertion. The specified tokens populate
2084 the [Endorsing Supporting Tokens] property.

2085 /sp:EndorsingSupportingTokens/wsp:Policy

2086 This describes additional requirements for satisfying the EndorsingSupportingTokens
2087 assertion.

2088 /sp:EndorsingSupportingTokens/wsp:Policy/[Token Assertion]

2089 The policy MUST identify one or more token assertions.

2090 /sp:EndorsingSupportingTokens/wsp:Policy/sp:AlgorithmSuite

2091 This optional element follows the schema outlined in Section 7.1 and describes the
2092 algorithms to use for cryptographic operations performed with the tokens identified
2093 by this policy assertion.

2094 /sp:EndorsingSupportingTokens/wsp:Policy/sp:SignedParts

2095 This optional element follows the schema outlined in Section 4.1.1 and describes
2096 additional message parts that MUST be included in the signature generated with the
2097 token identified by this policy assertion.

2098 /sp:EndorsingSupportingTokens/wsp:Policy/sp:SignedElements

2099 This optional element follows the schema outlined in Section 4.1.2 and describes
2100 additional message elements that MUST be included in the signature generated with
2101 the token identified by this policy assertion.

2102 /sp:EndorsingSupportingTokens/wsp:Policy/sp:EncryptedParts

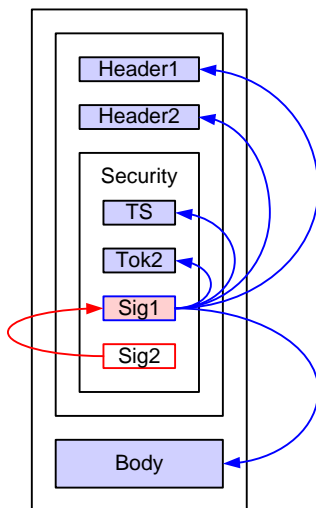
2103 This optional element follows the schema outlined in Section 4.2.1 and describes
2104 additional message parts that MUST be encrypted using the token identified by this
2105 policy assertion.

2106 /sp:EndorsingSupportingTokens/wsp:Policy/sp:EncryptedElements

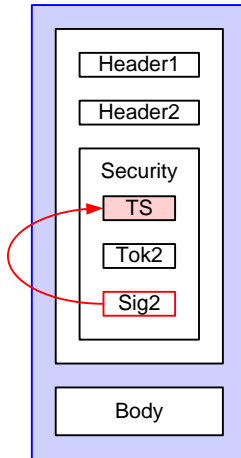
2107 This optional element follows the schema outlined in Section 4.2.2 and describes
2108 additional message elements that MUST be encrypted using the token identified by
2109 this policy assertion.

2110 8.4 SignedEndorsingSupportingTokens Assertion

2111 Signed endorsing tokens sign the entire `ds:Signature` element produced from the message
2112 signature and are themselves signed by that message signature, that is both tokens (the
2113 token used for the message signature and the signed endorsing token) sign each other. This
2114 assertion may optionally include additional message parts to sign and/or encrypt. The
2115 diagram below illustrates how the signed token (Tok2) is signed by the message signature
2116 (Sig1) and the endorsing signature (Sig2) signs the message signature (Sig1):
2117



2118
2119 If transport security is used, the token (Tok2) is included in the Security header and the
2120 signature (Sig2) should cover the message timestamp as illustrated below:
2121



2122

2123 **Syntax**

```

2124 <sp:SignedEndorsingSupportingTokens ... >
2125   <wsp:Policy>
2126     [Token Assertion]+
2127     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite ?
2128     (
2129       <sp:SignedParts ... > ... </sp:SignedParts> |
2130       <sp:SignedElements ... > ... </sp:SignedElements> |
2131       <sp:EncryptedParts ... > ... </sp:EncryptedParts> |
2132       <sp:EncryptedElements ... > ... </sp:EncryptedElements> |
2133     ) *
2134     ...
2135   </wsp:Policy>
2136   ...
2137 </sp:SignedEndorsingSupportingTokens>

```

2138

2139 The following describes the attributes and elements listed in the schema outlined above:

2140

/sp:SignedEndorsingSupportingTokens

2141

This identifies a SignedEndorsingSupportingTokens assertion. The specified tokens populate the [Signed Endorsing Supporting Tokens] property.

2142

2143 /sp:SignedEndorsingSupportingTokens/wsp:Policy

2144

This describes additional requirements for satisfying the EndorsingSupportingTokens assertion.

2145

2146 /sp:SignedEndorsingSupportingTokens/wsp:Policy/[Token Assertion]

2147

The policy MUST identify one or more token assertions.

2148 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:AlgorithmSuite

2149

This optional element follows the schema outlined in Section 7.1 and describes the algorithms to use for cryptographic operations performed with the tokens identified by this policy assertion.

2150

2151

2152 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:SignedParts

2153

This optional element follows the schema outlined in Section 4.1.1 and describes additional message parts that MUST be included in the signature generated with the token identified by this policy assertion.

2154

2155

2156 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:SignedElements

2157 This optional element follows the schema outlined in Section 4.1.2 and describes
2158 additional message elements that MUST be included in the signature generated with
2159 the token identified by this policy assertion.

2160 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:EncryptedParts

2161 This optional element follows the schema outlined in Section 4.2.1 and describes
2162 additional message parts that MUST be encrypted using the token identified by this
2163 policy assertion.

2164 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:EncryptedElements

2165 This optional element follows the schema outlined in Section 4.2.2 and describes
2166 additional message elements that MUST be encrypted using the token identified by
2167 this policy assertion.

2168 **8.5 Interaction between [Token Protection] property and** 2169 **supporting token assertions**

2170 If [Token Protection] (see Section 6.5) is true, then each signature covers the token that
2171 generated that signature and the following statements hold with respect to the various
2172 tokens that sign or are signed;

2173 The message signature, generated from the [Initiator Token] in the Asymmetric
2174 Binding case, or the [Signature Token] in the Symmetric binding case, covers that
2175 token.

2176 Endorsing signatures cover the main signature and the endorsing token.

2177 For signed, endorsing supporting tokens, the supporting token is signed twice, once
2178 by the message signature and once by the endorsing signature.

2179 In addition, signed supporting tokens are covered by the message signature, although this
2180 is independent of [Token Protection].

2181 **8.6 Example**

2182 Example policy containing supporting token assertions.

2183

```
<!-- Example Endpoint Policy -->
```



```

2184 <wsp:Policy>
2185   <sp:SymmetricBinding>
2186     <wsp:Policy>
2187       <sp:ProtectionToken>
2188         <sp:IssuedToken sp:IncludeToken="../../../IncludeToken/Once" >
2189           <sp:Issuer>...</sp:Issuer>
2190           <sp:RequestSecurityTokenTemplate>
2191             ...
2192           </sp:RequestSecurityTokenTemplate>
2193         </sp:IssuedToken>
2194       </sp:ProtectionToken>
2195       <sp:AlgorithmSuite>
2196         <wsp:Policy>
2197           <sp:Basic256 />
2198         </wsp:Policy>
2199       </sp:AlgorithmSuite>
2200       ...
2201     </wsp:Policy>
2202   </sp:SymmetricBinding>
2203   ...
2204   <sp:SignedSupportingTokens>
2205     <wsp:Policy>
2206       <sp:UsernameToken sp:IncludeToken="../../../IncludeToken/Once" />
2207     </wsp:Policy>
2208   </sp:SignedSupportingTokens>
2209   <sp:SignedEndorsingSupportingTokens>
2210     <wsp:Policy>
2211       <sp:X509V3Token sp:IncludeToken="../../../IncludeToken/Once" />
2212     </wsp:Policy>
2213   </sp:SignedEndorsingSupportingTokens>
2214   ...
2215 </wsp:Policy>

```

2216 The sp:SignedSupportingTokens assertion in the above policy indicates that a Username
2217 Token must be included in the security header and covered by the message signature. The
2218 sp:SignedEndorsingSupportingTokens assertion indicates that an X509 certificate must be
2219 included in the security header and covered by the message signature. In addition, a
2220 signature over the message signature based on the key material associated with the X509
2221 certificate must be included in the security header.

2222

9 WSS: SOAP Message Security Options

2223 There are several optional aspects to the WSS: SOAP Message Security specification that
2224 are independent of the trust and token taxonomies. This section describes another class of
2225 properties and associated assertions that indicate the supported aspects of WSS: SOAP
2226 Message Security. The assertions defined here MUST apply to [Endpoint Policy Subject].

2227 The properties and assertions dealing with token references defined in this section indicate
2228 whether the initiator and recipient MUST be able to process a given reference mechanism,
2229 or whether the initiator and recipient MAY send a fault if such references are encountered.

2230

2231 Note: This approach is chosen because:

2232 A) [WSS: SOAP Message Security] allows for multiple equivalent reference mechanisms
2233 to be used in a single reference.

2234 B) In a multi-message exchange, a token may be referenced using different
2235 mechanisms depending on which of a series of messages is being secured.

2236 **WSS: SOAP Message Security 1.0 Properties**

2237 **[Direct References]**

2238 This property indicates whether the initiator and recipient MUST be able to process direct
2239 token references (by ID or URI reference). This property always has a value of 'true'. i.e. All
2240 implementations MUST be able to process such references.

2241

2242 **[Key Identifier References]**

2243 This boolean property indicates whether the initiator and recipient MUST be able to process
2244 key-specific identifier token references. A value of 'true' indicates that the initiator and
2245 recipient MUST be able to generate and process such references. A value of 'false' indicates
2246 that the initiator and recipient MUST NOT generate such references and that the initiator
2247 and recipient MAY send a fault if such references are encountered. This property has a
2248 default value of 'false'.

2249

2250 **[Issuer Serial References]**

2251 This boolean property indicates whether the initiator and recipient MUST be able to process
2252 references using the issuer and token serial number. A value of 'true' indicates that the
2253 initiator and recipient MUST be able to process such references. A value of 'false' indicates
2254 that the initiator and recipient MUST NOT generate such references and that the initiator
2255 and recipient MAY send a fault if such references are encountered. This property has a
2256 default value of 'false'.

2257

2258 **[External URI References]**

2259 This boolean property indicates whether the initiator and recipient MUST be able to process
2260 references to tokens outside the message using URIs. A value of 'true' indicates that the
2261 initiator and recipient MUST be able to process such references. A value of 'false' indicates
2262 that the initiator and recipient MUST NOT generate such references and that the initiator
2263 and recipient MAY send a fault if such references are encountered. This property has a
2264 default value of 'false'.

2265 **[Embedded Token References]**

2266 This boolean property indicates whether the initiator and recipient MUST be able to process
2267 references that contain embedded tokens. A value of 'true' indicates that the initiator and
2268 recipient MUST be able to process such references. A value of 'false' indicates that the
2269 initiator and recipient MUST NOT generate such references and that the initiator and
2270 recipient MAY send a fault if such references are encountered. This property has a default
2271 value of 'false'.

2272

2273 **WSS: SOAP Message Security 1.1 Properties**

2274 **[Thumbprint References]**

2275 This boolean property indicates whether the initiator and recipient MUST be able to process
2276 references using token thumbprints. A value of 'true' indicates that the initiator and
2277 recipient MUST be able to process such references. A value of 'false' indicates that the
2278 initiator and recipient MUST NOT generate such references and that the initiator and
2279 recipient MAY send a fault if such references are encountered. This property has a default
2280 value of 'false'.

2281

2282 **[EncryptedKey References]**

2283 This boolean property indicates whether the initiator and recipient MUST be able to process
2284 references using EncryptedKey references. A value of 'true' indicates that the initiator and
2285 recipient MUST be able to process such references. A value of 'false' indicates that the
2286 initiator and recipient MUST NOT generate such references and that the initiator and
2287 recipient MAY send a fault if such references are encountered. This property has a default
2288 value of 'false'.

2289

2290 **[Signature Confirmation]**

2291 This boolean property specifies whether `wss11:SignatureConfirmation` elements should
2292 be used as defined in WSS: Soap Message Security 1.1. If the value is 'true',
2293 `wss11:SignatureConfirmation` elements MUST be used. If the value is 'false', signature
2294 confirmation elements MUST NOT be used. The value of this property applies to all
2295 signatures that are included in the security header. This property has a default value of
2296 'false'.

2297 **9.1 Wss10 Assertion**

2298 The Wss10 assertion allows you to specify which WSS: SOAP Message Security 1.0 options
2299 are supported.

2300 **Syntax**

```
2301 <sp:Wss10 ... >  
2302   <wsp:Policy>  
2303     <sp:MustSupportRefKeyIdentifier ... /> ?  
2304     <sp:MustSupportRefIssuerSerial ... /> ?  
2305     <sp:MustSupportRefExternalURI ... /> ?  
2306     <sp:MustSupportRefEmbeddedToken ... /> ?  
2307     ...  
2308   </wsp:Policy>  
2309   ...  
2310 </sp:Wss10>
```

2311

2312 The following describes the attributes and elements listed in the schema outlined above:
2313 /sp:Wss10
2314 This identifies a WSS10 assertion.
2315 /sp:Wss10/wsp:Policy
2316
2317 This indicates a policy that controls WSS: SOAP Message Security 1.0
2318 options./sp:Wss10/wsp:Policy/sp:MustSupportRefKeyIdentifier
2319 This assertion indicates that the [Key Identifier References] property is set to 'true'.
2320 /sp:Wss10/wsp:Policy/sp:MustSupportRefIssuerSerial
2321 This assertion indicates that the [Issuer Serial References] property is set to 'true'.
2322 /sp:Wss10/wsp:Policy/sp:MustSupportRefExternalURI
2323 This assertion indicates that the [External URI References] property is set to 'true'.
2324 /sp:Wss10/wsp:Policy/sp:MustSupportRefEmbeddedToken
2325 This assertion indicates that the [Embedded Token References] property is set to
2326 'true'.

2327 9.2 Wss11 Assertion

2328 The Wss11 assertion allows you to specify which WSS: SOAP Message Security 1.1 options
2329 are supported.

2330 Syntax

```
2331 < sp:Wss11 ... >  
2332 <wsp:Policy>  
2333 <sp:MustSupportRefKeyIdentifier ... /> ?  
2334 <sp:MustSupportRefIssuerSerial ... /> ?  
2335 <sp:MustSupportRefExternalURI ... /> ?  
2336 <sp:MustSupportRefEmbeddedToken ... /> ?  
2337 <sp:MustSupportRefThumbprint ... /> ?  
2338 <sp:MustSupportRefEncryptedKey ... /> ?  
2339 <sp:RequireSignatureConfirmation ... /> ?  
2340 ...  
2341 </wsp:Policy>  
2342 </sp:Wss11>
```

2343
2344 The following describes the attributes and elements listed in the schema outlined above:
2345 /sp:Wss11
2346 This identifies an WSS11 assertion.
2347 /sp:Wss11/wsp:Policy
2348 This indicates a policy that controls WSS: SOAP Message Security 1.1 options.
2349 /sp:Wss11/wsp:Policy/sp:MustSupportRefKeyIdentifier
2350 This assertion indicates that the [Key Identifier References] property is set to 'true'.
2351 /sp:Wss11/wsp:Policy/sp:MustSupportRefIssuerSerial
2352 This assertion indicates that the [Issuer Serial References] property is set to 'true'.
2353 /sp:Wss11/wsp:Policy/sp:MustSupportRefExternalURI
2354 This assertion indicates that the [External URI References] property is set to 'true'.

2355 /sp:Wss11/wsp:Policy/sp:MustSupportRefEmbeddedToken
2356 This assertion indicates that the [Embedded Token References] property is set to
2357 'true'.
2358 /sp:Wss11/wsp:Policy/sp:MustSupportRefThumbprint
2359 This assertion indicates that the [Thumbprint References] property is set to 'true'.
2360 /sp:Wss11/wsp:Policy/sp:MustSupportRefEncryptedKey
2361 This assertion indicates that the [EncryptedKey References] property is set to 'true'.
2362 /sp:Wss11/wsp:Policy/sp:RequireSignatureConfirmation
2363 This assertion indicates that the [Signature Confirmation] property is set to 'true'.

2364 **10 WS-Trust Options**

2365 This section defines the various policy assertions related to exchanges based on WS-Trust,
2366 specifically with client and server challenges and entropy behaviors. These assertions relate
2367 to interactions with a Security Token Service and may augment the behaviors defined by
2368 the Binding Property Assertions defined in Section 6. The assertions defined here MUST
2369 apply to [Endpoint Policy Subject].

2370

2371 **WS-Trust 1.0 Properties**

2372 **[Client Challenge]**

2373 This boolean property indicates whether client challenges are supported. A value of 'true'
2374 indicates that a `wst:SignChallenge` element is supported inside of an RST sent by the client
2375 to the server. A value of 'false' indicates that a `wst:SignChallenge` is not supported. There is
2376 no change in the number of messages exchanged by the client and service in satisfying the
2377 RST. This property has a default value of 'false'.

2378

2379 **[Server Challenge]**

2380 This boolean property indicates whether server challenges are supported. A value of 'true'
2381 indicates that a `wst:SignChallenge` element is supported inside of an RSTR sent by the
2382 server to the client. A value of 'false' indicates that a `wst:SignChallenge` is not supported. A
2383 challenge issued by the server may increase the number of messages exchanged by the
2384 client and service in order to accommodate the `wst:SignChallengeResponse` element sent by
2385 the client to the server in response to the `wst:SignChallenge` element. A final RSTR
2386 containing the issued token will follow subsequent to the server receiving the
2387 `wst:SignChallengeResponse` element. This property has a default value of 'false'.

2388

2389 **[Client Entropy]**

2390 This boolean property indicates whether client entropy is required to be used as key
2391 material for a requested proof token. A value of 'true' indicates that client entropy is
2392 required. A value of 'false' indicates that client entropy is not required. This property has a
2393 default value of 'false'.

2394

2395 **[Server Entropy]**

2396 This boolean property indicates whether server entropy is required to be used as key
2397 material for a requested proof token. A value of 'true' indicates that server entropy is
2398 required. A value of 'false' indicates that server entropy is not required. This property has a
2399 default value of 'false'.

2400 Note: If both the [Client Entropy] and [Server Entropy] properties are set to true, Client and
2401 server entropy are combined to produce a computed key using the Computed Key algorithm
2402 defined by the [Algorithm Suite] property.

2403

2404 **[Issued Tokens]**

2405 This boolean property indicates whether the `wst:IssuedTokens` header is supported as
2406 described in WS-Trust. A value of 'true' indicates that the `wst:IssuedTokens` header is

2407 supported. A value of 'false' indicates that the `wst:IssuedTokens` header is not supported.
2408 This property has a default value of 'false'.

2409 **10.1 Trust10 Assertion**

2410 The Trust10 assertion allows you to specify which WS-Trust 1.0 options are supported.

2411 **Syntax**

```
2412 <sp:Trust10 ... >  
2413 <wsp:Policy>  
2414 <sp:MustSupportClientChallenge ... />?  
2415 <sp:MustSupportServerChallenge ... />?  
2416 <sp:RequireClientEntropy ... />?  
2417 <sp:RequireServerEntropy ... />?  
2418 <sp:MustSupportIssuedTokens ... />?  
2419 ...  
2420 </wsp:Policy>  
2421 ...  
2422 </sp:Trust10 ... >
```

2423
2424 The following describes the attributes and elements listed in the schema outlined above:

2425 `/sp:Trust10`

2426 This identifies a Trust10 assertion.

2427 `/sp:Trust10/wsp:Policy`

2428 This indicates a policy that controls WS-Trust 1.0 options.

2429 `/sp:Trust10/wsp:Policy/sp:MustSupportClientChallenge`

2430 This assertion indicates that the [Client Challenge] property is set to 'true'.

2431 `/sp:Trust10/wsp:Policy/sp:MustSupportServerChallenge`

2432 This assertion indicates that the [Server Challenge] property is set to 'true'.

2433 `/sp:Trust10/wsp:Policy/sp:RequireClientEntropy`

2434 This assertion indicates that the [Client Entropy] property is set to 'true'.

2435 `/sp:Trust10/wsp:Policy/sp:RequireServerEntropy`

2436 This assertion indicates that the [Server Entropy] property is set to 'true'.

2437 `/sp:Trust10/wsp:Policy/sp:MustSupportIssuedTokens`

2438 This assertion indicates that the [Issued Tokens] property is set to 'true'.

2439 **11 Guidance on creating new assertions and** 2440 **assertion extensibility**

2441 This non-normative appendix provides guidance for designers of new assertions intended for
2442 use with this specification.

2443 **11.1 General Design Points**

2444 Prefer Distinct Qnames

2445 Parameterize using nested policy where possible.

2446 Parameterize using attributes and/or child elements where necessary.

2447 **11.2 Detailed Design Guidance**

2448 Assertions in WS-SP are XML elements that are identified by their QName. Matching of
2449 assertions per WS-Policy is performed by matching element QNames. Matching does not
2450 take into account attributes that are present on the assertion element. Nor does it take into
2451 account child elements except for wsp:Policy elements. If a wsp:Policy element is present,
2452 then matching occurs against the assertions nested inside that wsp:Policy element
2453 recursively (see Section 3.1).

2454

2455 When designing new assertions for use with WS-SP, the above matching behaviour needs to
2456 be taken into account. In general, multiple assertions with distinct QNames are preferably
2457 to a single assertion that uses attributes and/or content to distinguish different cases. For
2458 example, given two possible assertion designs;

2459

2460

2461

2462

2463

2464

2465

2466

2467

2468

2469

2470

2471

```
Design 1
  <A1/>
  <A2/>
  <A3/>

Design 2.
  <A Parameter='1' />
  <A Parameter='2' />
  <A Parameter='3' />
```

2472 then design 1. would generally be preferred because it allows the policy matching logic to
2473 provide more accurate matches between policies.

2474

2475 A good example of design 1. is the token assertions defined in Section 5. The section
2476 defines 10 distinct token assertions, rather than a single sp:Token assertion with, for
2477 example, a TokenType attribute. These distinct token assertions make policy matching
2478 much more useful as less false positives are generated when performing policy matching.

2479

2480 There are cases where using attributes or child elements as parameters in assertion design
2481 is reasonable. Examples include cases when implementations are expected to understand all

2482 the values for a given parameter and when encoding the parameter information into the
2483 assertion QName would result in an unmanageable number of assertions. A good example is
2484 the sp:IncludeToken attribute that appears on the various token assertions. Five possible
2485 values are currently specified for the sp:IncludeToken attribute and implementations are
2486 expected to understand the meaning of all 5 values. If this information was encoded into
2487 the assertion QNames, each existing token assertion would require five variants, one for
2488 each Uri value which would result in 45 assertions just for the tokens defined in Section 5.

2489

2490 Nested policy is ideal for encoding parameters that can be usefully matched using policy
2491 matching. For example, the token version assertions defined in Section 5 use such an
2492 approach. The overall token type assertion is parameterized by the nested token version
2493 assertions. Policy matching can use these parameters to find matches between policies
2494 where the broad token type is support by both parties but they might not support the same
2495 specific versions.

2496

2497 Note, when designing assertions for new token types such assertions SHOULD allow the
2498 sp:IncludeToken attribute and SHOULD allow nested policy.

2499

2500 **12 Security Considerations**

- 2501 It is strongly recommended that policies and assertions be signed to prevent tampering.
- 2502 It is recommended that policies should not be accepted unless they are signed and have an
2503 associated security token to specify the signer has proper claims for the given policy. That
2504 is, a party shouldn't rely on a policy unless the policy is signed and presented with sufficient
2505 claims. It is further recommended that the entire policy exchange mechanism be protected
2506 to prevent man-in-the-middle downgrade attacks.
- 2507
- 2508 It should be noted that the mechanisms described in this document could be secured as
2509 part of a SOAP message using [WSS: SOAP Message Security](#) or embedded within other
2510 objects using object-specific security mechanisms.
- 2511
- 2512 It is recommended that policies not specify two (or more) SignedSupportingTokens or
2513 SignedEndorsingSupportingTokens of the same token type. Messages conforming to such
2514 policies are subject to modification which may be undetectable.
- 2515
- 2516 It is recommended that policies specify the OnlySignEntireHeadersAndBody assertion along
2517 with the rest of the policy in order to combat certain XML substitution attacks.

2518 **A. Assertions and WS-PolicyAttachment**

2519 This non-normative appendix classifies assertions according to their suggested scope in
2520 WSDL 1.1 per Section 4 of [WS-PolicyAttachment]. See Figure 1 in Section 4.1 of [WS-
2521 PolicyAttachment] for a graphical representation of the relationship between policy scope
2522 and WSDL. Unless otherwise noted above, any assertion that is listed under multiple [Policy
2523 Subjects] below MUST only apply to only one [Policy Subject] in a WSDL 1.1 hierarchy for
2524 calculating an Effective Policy

2525 **A.1 Endpoint Policy Subject Assertions**

2526 **A.1.1 Security Binding Assertions**

2527 TransportBinding Assertion (Section 7.3)
2528 SymmetricBinding Assertion (Section 7.4)
2529 AsymmetricBinding Assertion (Section 7.5)

2530 **A.1.2 Token Assertions**

2531 SupportingTokens Assertion (Section 8.1)
2532 SignedSupportingTokens Assertion (Section 8.2)
2533 EndorsingSupportingTokens Assertion (Section 8.3)
2534 SignedEndorsingSupportingTokens Assertion (Section 8.4)

2535 **A.1.3 WSS: SOAP Message Security 1.0 Assertions**

2536 Wss10 Assertion (Section 9.1)

2537 **A.1.4 WSS: SOAP Message Security 1.1 Assertions**

2538 Wss11 Assertion (Section 9.2)

2539 **A.1.5 Trust 1.0 Assertions**

2540 Trust10 Assertion (Section 10.1)

2541 **A.2 Operation Policy Subject Assertions**

2542 **A.2.1 Security Binding Assertions**

2543 SymmetricBinding Assertion (Section 7.4)
2544 AsymmetricBinding Assertion (Section 7.5)

2545 **A.2.2 Supporting Token Assertions**

2546 SupportingTokens Assertion (Section 8.1)
2547 SignedSupportingTokens Assertion (Section 8.2)
2548 EndorsingSupportingTokens Assertion (Section 8.3)

2549 SignedEndorsingSupportingTokens Assertion (Section 8.4)

2550 **A.3 Message Policy Subject Assertions**

2551 **A.3.1 Supporting Token Assertions**

2552 SupportingTokens Assertion (Section 8.1)

2553 SignedSupportingTokens Assertion (Section 8.2)

2554 EndorsingSupportingTokens Assertion (Section 8.3)

2555 SignedEndorsingSupportingTokens Assertion (Section 8.4)

2556 **A.3.2 Protection Assertions**

2557 SignedParts Assertion (Section 4.1.1)

2558 SignedElements Assertion (Section 4.1.2)

2559 EncryptedParts Assertion (Section 4.2.1)

2560 EncryptedElements Assertion (Section 4.2.2)

2561 RequiredElements Assertion (Section 4.3.1)

2562 **A.4 Assertions With Undefined Policy Subject**

2563 The assertions listed in this section do not have a defined policy subject because they
2564 appear nested inside some other assertion which does have a defined policy subject.

2565 **A.4.1 General Assertions**

2566 AlgorithmSuite Assertion (Section 7.1)

2567 Layout Assertion (Section 7.2)

2568 IncludeTimestamp Assertion (Section 7.3)

2569 IncludeTimestamp Assertion (Section 7.4)

2570 EncryptBeforeSigning Assertion (Section 7.4)

2571 EncryptSignature Assertion (Section 7.4)

2572 ProtectTokens Assertion (Section 7.4)

2573 OnlySignEntireHeadersAndBody Assertion (Section 7.4)

2574 IncludeTimestamp Assertion (Section 7.5)

2575 EncryptBeforeSigning Assertion (Section 7.5)

2576 EncryptSignature Assertion (Section 7.5)

2577 ProtectTokens Assertion (Section 7.5)

2578 OnlySignEntireHeadersAndBody Assertion (Section 7.5)

2579 **A.4.2 Token Usage Assertions**

2580 TransportToken Assertion (Section 7.3)

2581 EncryptionToken Assertion (Section 7.4)

2582 SignatureToken Assertion (Section 7.4)

2583 ProtectionToken Assertion (Section 7.4)

2584 InitiatorToken Assertion (Section 7.5)

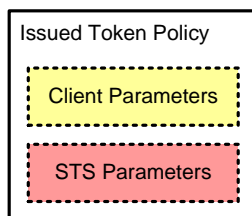
2585	RecipientToken Assertion	(Section 7.5)
2586	A.4.3 Token Assertions	
2587	UsernameToken Assertion	(Section 5.3.1)
2588	IssuedToken Assertion	(Section 5.3.2)
2589	X509Token Assertion	(Section 5.3.3)
2590	KerberosToken Assertion	(Section 5.3.4)
2591	SpnegoContextToken Assertion	(Section 5.3.5)
2592	SecurityContextToken Assertion	(Section 5.3.6)
2593	SecureConversationToken Assertion	(Section 5.3.7)
2594	SamlToken Assertion	(Section 5.3.8)
2595	RelToken Assertion	(Section 5.3.9)
2596	HttpsToken Assertion	(Section 5.3.10)
2597	A.4.4 WSS: SOAP Message Security 1.0 Assertions	
2598	MustSupportRefKeyIdentifier Assertion	(Section 9.1)
2599	MustSupportRefIssuerSerial Assertion	(Section 9.1)
2600	MustSupportRefExternalUri Assertion	(Section 9.1)
2601	MustSupportRefEmbeddedToken Assertion	(Section 9.1)
2602	A.4.5 WSS: SOAP Message Security 1.1 Assertions	
2603	MustSupportRefKeyIdentifier Assertion	(Section 9.2)
2604	MustSupportRefIssuerSerial Assertion	(Section 9.2)
2605	MustSupportRefExternalUri Assertion	(Section 9.2)
2606	MustSupportRefEmbeddedToken Assertion	(Section 9.2)
2607	MustSupportRefThumbprint Assertion	(Section 9.2)
2608	RequireSignatureConfirmation Assertion	(Section 9.2)
2609	A.4.6 Trust 1.0 Assertions	
2610	MustSupportClientChallenge Assertion	(Section 10.1)
2611	MustSupportServerChallenge Assertion	(Section 10.1)
2612	RequireClientEntropy Assertion	(Section 10.1)
2613	RequireServerEntropy Assertion	(Section 10.1)
2614	MustSupportIssuedTokens Assertion	(Section 10.1)

2615 B.Issued Token Policy

2616 The section provides further detail about behavior associated with the IssuedToken
2617 assertion in section 5.3.2.

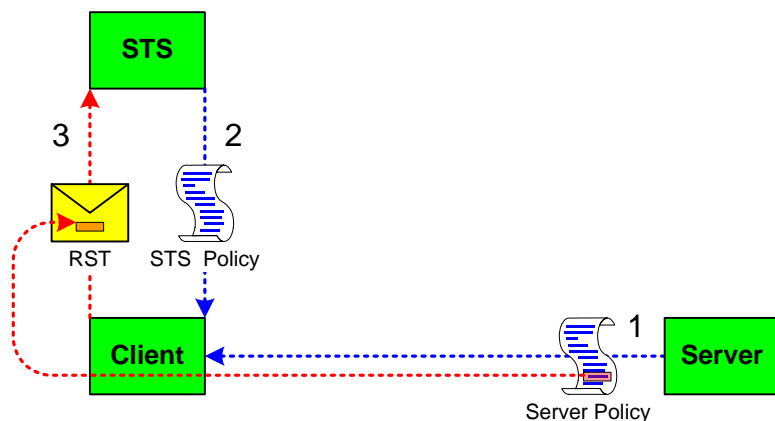
2618
2619 The issued token security model involves a three-party setup. There's a target Server, a
2620 Client, and a trusted third party called a Security Token Service or STS. Policy flows from
2621 Server to Client, and from STS to Client. Policy may be embedded inside an Issued Token
2622 assertion, or acquired out-of-band. There may be an explicit trust relationship between the
2623 Server and the STS. There must be a trust relationship between the Client and the STS.

2624
2625 The Issued Token policy assertion includes two parts: 1) client-specific parameters that
2626 must be understood and processed by the client and 2) STS specific parameters which are
2627 to be processed by the STS. The format of the Issued Token policy assertion is illustrated in
2628 the figure below.



2629
2630 The client-specific parameters of the Issued Token policy assertion along with the remainder
2631 of the server policy are consumed by the client. The STS specific parameters of the Issued
2632 Token policy assertion are passed on to the STS by copying the parameters directly into the
2633 RST request sent by the Client to the STS as illustrated in the figure below.

2634



2635
2636 Before the Client sends the RST to the STS, it will need to obtain the policy for the STS. This
2637 will help to formulate the RST request and will include any security-specific requirements of
2638 the STS.

2639

2640 The Client may augment or replace the contents of the RST made to the STS based on the
2641 Client-specific parameters received from the Issued Token policy assertion contained in the
2642 Server policy, from policy it received for the STS, or any other local parameters.

2643

2644 The Issued Token Policy Assertion contains elements which must be understood by the
2645 Client. The assertion contains one element which contains a list of arbitrary elements which
2646 should be sent along to the STS by copying the elements as-is directly into the request sent
2647 by the Client to the STS following the protocol defined in WS-Trust.

2648

2649 Elements inside the `sp:RequestSecurityTokenTemplate` element MUST conform to WS-
2650 Trust [WS-Trust]. All items are optional, since the Server and STS may already have a pre-
2651 arranged relationship which specifies some or all of the conditions and constraints for issued
2652 tokens.

2653 C. Strict Security Header Layout Examples

2654 The following sections describe the security header layout for specific bindings when
2655 applying the 'Strict' layout rules defined in Section 6.7.

2656 C.1 Transport Binding

2657 This section describes how the 'Strict' security header layout rules apply to the Transport
2658 Binding.

2659 C.1.1 Policy

2660 The following example shows a policy indicating a Transport Binding, an Https Token as the
2661 Transport Token, an algorithm suite, a requirement to include tokens in the supporting
2662 signatures, a username token attached to the message, and finally an X509 token attached
2663 to the message and endorsing the message signature. No message protection requirements
2664 are described since the transport covers all message parts.

```
2665 <wsp:Policy>  
2666   <sp:TransportBinding>  
2667     <wsp:Policy>  
2668       <sp:TransportToken>  
2669         <wsp:Policy>  
2670           <sp:HttpsToken />  
2671         </wsp:Policy>  
2672       </sp:TransportToken>  
2673       <sp:AlgorithmSuite>  
2674         <wsp:Policy>  
2675           <sp:Basic256 />  
2676         </wsp:Policy>  
2677       </sp:AlgorithmSuite>  
2678       <sp:Layout>  
2679         <wsp:Policy>  
2680           <sp:Strict />  
2681         </wsp:Policy>  
2682       </sp:Layout>  
2683       <sp:IncludeTimestamp />  
2684     </wsp:Policy>  
2685   </sp:TransportBinding>  
2686   <sp:SignedSupportingTokens>  
2687     <wsp:Policy>  
2688       <sp:UsernameToken sp:IncludeToken="../../../IncludeToken/Once" />  
2689     </wsp:Policy>  
2690   </sp:SignedSupportingTokens>  
2691   <sp:SignedEndorsingSupportingTokens>  
2692     <wsp:Policy>  
2693       <sp:X509V3Token sp:IncludeToken="../../../IncludeToken/Once" />  
2694     </wsp:Policy>  
2695   </sp:SignedEndorsingSupportingTokens>  
2696   <sp:Wss11>  
2697     <sp:RequireSignatureConfirmation />  
2698   </sp:Wss11>  
2699 </wsp:Policy>
```

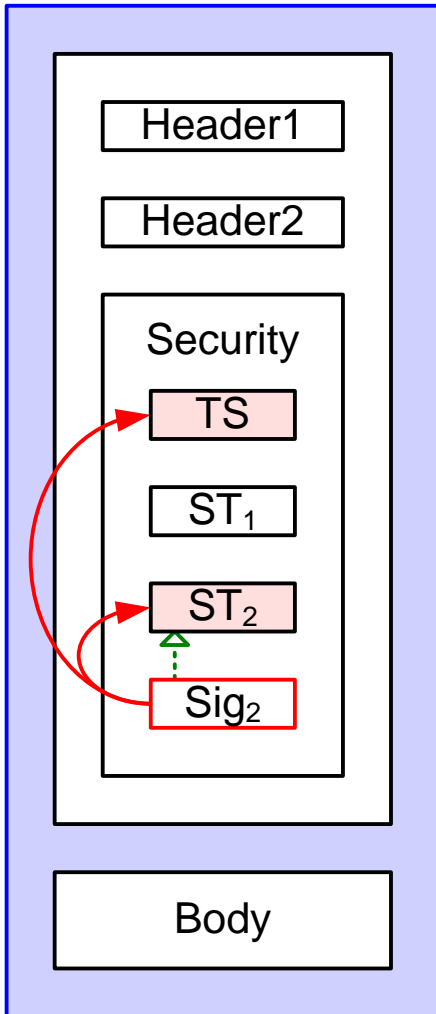
2700 This policy is used as the basis for the examples shown in the subsequent section describing
2701 the security header layout for this binding.

2702 **C.1.2 Initiator to Recipient Messages**

2703 Messages sent from initiator to recipient have the following layout for the security header:

- 2704 1. A `wsu:Timestamp` element.
- 2705 2. Any tokens contained in the [Signed Supporting Tokens] property.
- 2706 3. Any tokens contained in the [Signed Endorsing Supporting Tokens] property each
2707 followed by the corresponding signature. Each signature MUST cover the
2708 `wsu:Timestamp` element from 1 above and SHOULD cover any other unique identifier
2709 for the message in order to prevent replays. If [Token Protection] is 'true', the
2710 signature MUST also cover the supporting token. If [Derived Keys] is 'true' and the
2711 supporting token is associated with a symmetric key, then a Derived Key Token,
2712 based on the supporting token, appears between the supporting token and the
2713 signature.
- 2714 4. Any signatures for tokens contained in the [Endorsing Supporting Tokens] property.
2715 Each signature MUST cover the `wsu:Timestamp` element from 1 above and SHOULD
2716 cover at least some other unique identifier for the message in order to prevent
2717 replays. If [Token Protection] is 'true', the signature MUST also cover the supporting
2718 token. If [Derived Keys] is 'true' and the supporting token is associated with a
2719 symmetric key, then a Derived Key Token, based on the supporting token, appears
2720 before the signature.

2721 The following diagram illustrates the security header layout for the initiator to recipient
2722 message:



2723

2724 The blue outer box shows that the entire message is protected (signed and encrypted) by
 2725 the transport. The red arrows (left) from the box labeled Sig₂ indicate the parts signed by
 2726 the supporting token labeled ST₂, namely the message timestamp labeled TS and the token
 2727 used as the basis for the signature labeled ST₂. The green dotted arrow indicates the token
 2728 that was used as the basis for the signature. In general, the ordering of the items in the
 2729 security header follows the most optimal layout for a receiver to process its contents.

2730 *Example:*

2731 Initiator to recipient message

```

2732 <S:Envelope>
2733   <S:Header>
2734     ...
2735     <wsse:Security>
2736       <wsu:Timestamp wsu:Id="timestamp">
2737         <wsu:Created>[datetime]</wsu:Created>
2738         <wsu:Expires>[datetime]</wsu:Expires>
2739       </wsu:Timestamp>
2740       <wsse:UsernameToken wsu:Id='SomeSignedToken' >
2741         ...
2742       </wsse:UsernameToken>
2743       <wsse:BinarySecurityToken wsu:Id="SomeSignedEndorsingToken" >
2744         ...
2745       </wsse:BinarySecurityToken>
2746       <ds:Signature>
2747         <ds:SignedInfo>
2748           <ds:References>
2749             <ds:Reference URI="#timestamp" />
2750             <ds:Reference URI="#SomeSignedEndorsingToken" />
2751           </ds:References>
2752         </ds:SignedInfo>
2753         <ds:SignatureValue>...</ds:SignatureValue>
2754         <ds:KeyInfo>
2755           <wsse:SecurityTokenReference>
2756             <wsse:Reference URI="#SomeSignedEndorsingToken" />
2757           </wsse:SecurityTokenReference>
2758         </ds:KeyInfo>
2759       </ds:Signature>
2760     ...
2761   </wsse:Security>
2762   ...
2763 </S:Header>
2764 <S:Body>
2765   ...
2766 </S:Body>
2767 </S:Envelope>

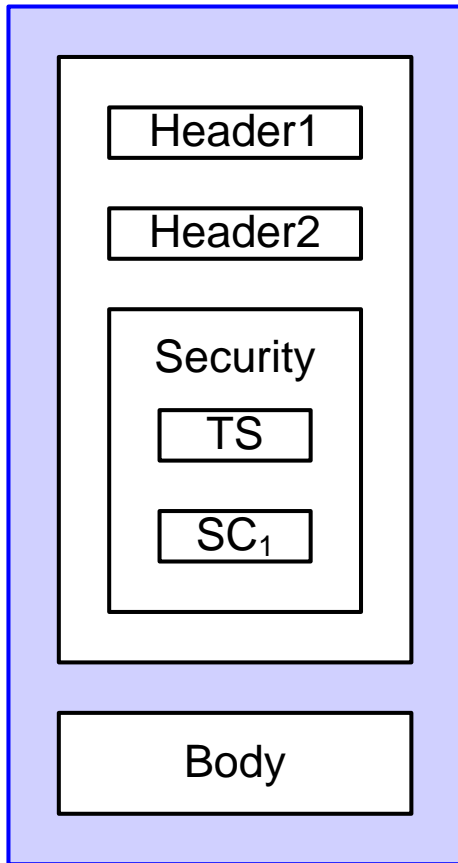
```

2768 **C.1.3 Recipient to Initiator Messages**

2769 Messages sent from recipient to initiator have the following layout for the security header:

- 2770 1. A `wsu:Timestamp` element.
- 2771 2. If the [Signature Confirmation] property has a value of 'true', then a
- 2772 `wsse11:SignatureConfirmation` element for each signature in the corresponding
- 2773 message sent from initiator to recipient. If there are no signatures in the
- 2774 corresponding message from the initiator to the recipient, then a
- 2775 `wsse11:SignatureConfirmation` element with no Value attribute.

2776 The following diagram illustrates the security header layout for the recipient to initiator
 2777 message:



2778

2779 The blue outer box shows that the entire message is protected (signed and encrypted) by
 2780 the transport. One `wsse11:SignatureConfirmation` element labeled `SC1` corresponding to
 2781 the signature in the initial message illustrated previously is included. In general, the
 2782 ordering of the items in the security header follows the most optimal layout for a receiver to
 2783 process its contents.

2784 *Example:*

2785 Recipient to initiator message

```

2786 <S:Envelope>
2787 <S:Header>
2788   ...
2789 <wsse:Security>
2790   <wsu:Timestamp wsu:Id="timestamp">
2791     <wsu:Created>[datetime]</wsu:Created>
2792     <wsu:Expires>[datetime]</wsu:Expires>
2793   </wsu:Timestamp>
2794   <wsse11:SignatureConfirmation Value="..." />
2795   ...
2796 </wsse:Security>
2797   ...
2798 </S:Header>
2799 <S:Body>
2800   ...
2801 </S:Body>
2802 </S:Envelope>
  
```

2803 **C.2 Symmetric Binding**

2804 This section describes how the 'Strict' security header layout rules apply to the Symmetric
2805 Binding.

2806 **C.2.1 Policy**

2807 The following example shows a policy indicating a Symmetric Binding, a symmetric key
2808 based IssuedToken provided as the Protection Token, an algorithm suite, a requirement to
2809 encrypt the message parts before signing, a requirement to encrypt the message signature,
2810 a requirement to include tokens in the message signature and the supporting signatures, a
2811 username token attached to the message, and finally an X509 token attached to the
2812 message and endorsing the message signature. Minimum message protection requirements
2813 are described as well.

```

2814 <!-- Example Endpoint Policy -->
2815 <wsp:Policy>
2816   <sp:SymmetricBinding>
2817     <wsp:Policy>
2818       <sp:ProtectionToken>
2819         <sp:IssuedToken sp:IncludeToken="../../../IncludeToken/Once" >
2820           <sp:Issuer>...</sp:Issuer>
2821           <sp:RequestSecurityTokenTemplate>
2822             ...
2823           </sp:RequestSecurityTokenTemplate>
2824         </sp:IssuedToken>
2825       </sp:ProtectionToken>
2826       <sp:AlgorithmSuite>
2827         <wsp:Policy>
2828           <sp:Basic256 />
2829         </wsp:Policy>
2830       </sp:AlgorithmSuite>
2831       <sp:Layout>
2832         <wsp:Policy>
2833           <sp:Strict />
2834         </wsp:Policy>
2835       </sp:Layout>
2836       <sp:IncludeTimestamp />
2837       <sp:EncryptBeforeSigning />
2838       <sp:EncryptSignature />
2839       <sp:ProtectTokens />
2840     </wsp:Policy>
2841   </sp:SymmetricBinding>
2842   <sp:SignedSupportingTokens>
2843     <wsp:Policy>
2844       <sp:UsernameToken sp:IncludeToken="../../../IncludeToken/Once" />
2845     </wsp:Policy>
2846   </sp:SignedSupportingTokens>
2847   <sp:SignedEndorsingSupportingTokens>
2848     <wsp:Policy>
2849       <sp:X509V3Token sp:IncludeToken="../../../IncludeToken/Once" />
2850     </wsp:Policy>
2851   </sp:SignedEndorsingSupportingTokens>
2852   <sp:Wss11>
2853     <wsp:Policy>
2854       <sp:RequireSignatureConfirmation />
2855     </wsp:Policy>
2856   </sp:Wss11>
2857 </wsp:Policy>
2858
2859 <!-- Example Message Policy -->
2860 <wsp:Policy>
2861   <sp:SignedParts>
2862     <sp:Header Name="Header1" Namespace="..." />
2863     <sp:Header Name="Header2" Namespace="..." />
2864     <sp:Body/>
2865   </sp:SignedParts>
2866   <sp:EncryptedParts>
2867     <sp:Header Name="Header2" Namespace="..." />
2868     <sp:Body/>
2869   </sp:EncryptedParts>
2870 </wsp:Policy>

```

2871 This policy is used as the basis for the examples shown in the subsequent section describing
2872 the security header layout for this binding.

2873 C.2.2 Initiator to Recipient Messages

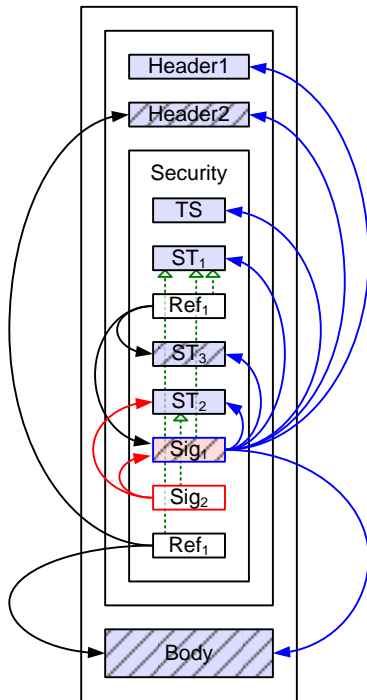
2874 Messages sent from initiator to recipient have the following layout for the security header:

- 2875 1. A `wsu:Timestamp` element if [Timestamp] is 'true'.
- 2876 2. If the `sp:IncludeToken` attribute on the [Encryption Token] is `.../IncludeToken/Once`
2877 or `.../IncludeToken/Always`, then the [Encryption Token].
- 2878 3. If [Derived Keys] is 'true', then a Derived Key Token, based on the [Encryption
2879 Token]. This Derived Key Token is used for encryption.
- 2880 4. A reference list including references to encrypted items. If [Signature Protection] is
2881 'true', then the reference list MUST include a reference to the message signature. If
2882 [Protection Order] is 'SignBeforeEncrypting', then the reference list MUST include a
2883 reference to all the message parts specified in the EncryptedParts assertions in the
2884 policy. If [Derived Keys] is 'true', then the key in the token from 3 above MUST be
2885 used, otherwise the key in the [Encryption Token].
- 2886 5. Any tokens from the [Signed Supporting Tokens] and [Signed Endorsing Supporting
2887 Tokens] properties whose `sp:IncludeToken` attribute is `.../IncludeToken/Once` or
2888 `.../IncludeToken/Always`.
- 2889 6. If the [Signature Token] is not the same as the [Encryption Token], and the
2890 `sp:IncludeToken` attribute on the [Signature Token] is `.../IncludeToken/Once` or
2891 `.../IncludeToken/Always`, then the [Signature Token].
- 2892 7. If [Derived Keys] is 'true', then a Derived Key Token based on the [Signature
2893 Token]. This Derived Key Token is used for signature.
- 2894 8. A signature over the `wsu:Timestamp` from 1 above, any tokens from 5 above
2895 regardless of whether they are included in the message, and any message parts
2896 specified in SignedParts assertions in the policy. If [Token Protection] is 'true', the
2897 signature MUST cover the [Signature Token] regardless of whether it is included in
2898 the message. If [Derived Keys] is 'true', the key in the token from 7 above MUST be
2899 used, otherwise the key in the [Signature Token] from 6 above.
- 2900 9. Signatures covering the main signature from 8 above for any tokens from the
2901 [Endorsing Supporting Tokens] and [Signed Endorsing Supporting Tokens]
2902 properties. If [Token Protection] is 'true', the signature MUST also cover the
2903 endorsing token. If [Derived Keys] is 'true' and the endorsing token is associated
2904 with a symmetric key, then a Derived Key Token, based on the endorsing token,
2905 appears before the signature.
- 2906 10. If [Protection Order] is 'EncryptBeforeSigning', then a reference list referencing all
2907 the message parts specified in EncryptedParts assertions in the policy. If [Derived
2908 Keys] is 'true', then the key in the token from 3 above MUST be used, otherwise the
2909 key in the [Encryption Token] from 2 above.

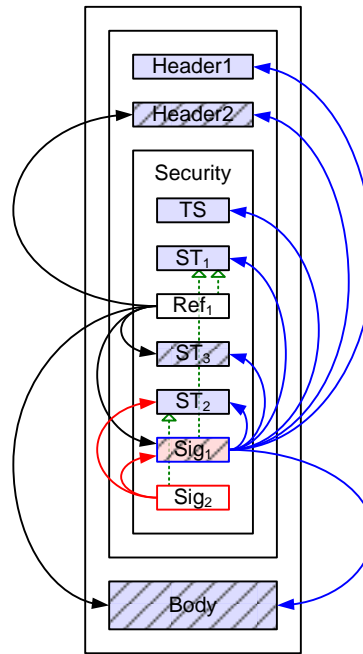
2910

2911 The following diagram illustrates the security header layout for the initiator to recipient
2912 message:

Encrypt Then Sign



Sign Then Encrypt



2913

2914 The blue arrows (right) indicate parts that were signed as part of the message signature
2915 labeled Sig₁. The red arrows (left) from the box labeled Sig₂ indicate the parts signed by the
2916 supporting token labeled ST₂, namely the message signature labeled Sig₁ and the token
2917 used as the basis for the signature labeled ST₂. The black arrows (left) from boxes labeled
2918 Ref₁ indicate references to parts encrypted using a key based on the Shared Secret Token
2919 labeled ST₁. The green dotted arrows indicate the token that was used as the basis for each
2920 cryptographic operation. In general, the ordering of the items in the security header follows
2921 the most optimal layout for a receiver to process its contents.

2922 *Example:*

2923 Initiator to recipient message using EncryptBeforeSigning.


```

2924 <S:Envelope>
2925   <S:Header>
2926     <x:Header1 wsu:Id="Header1" >
2927       ...
2928     </x:Header1>
2929     <wsse1:EncryptedHeader wsu:Id="enc_Header2">
2930       <!-- Plaintext Header2
2931       <x:Header2 wsu:Id="Header2" >
2932         ...
2933       </x:Header2>
2934       -->
2935     </wsse1:EncryptedHeader>
2936     ...
2937   <wsse:Security>
2938     <wsu:Timestamp wsu:Id="Timestamp">
2939       <wsu:Created>...</wsu:Created>
2940       <wsu:Expires>...</wsu:Expires>
2941     </wsu:Timestamp>
2942     <saml:Assertion AssertionId="_SharedSecretToken" ...>
2943       ...
2944     </saml:Assertion>
2945     <xenc:ReferenceList>
2946       <xenc:DataReference URI="#enc_Signature" />
2947       <xenc:DataReference URI="#enc_SomeUsernameToken" />
2948       ...
2949     </xenc:ReferenceList>
2950     <xenc:EncryptedData ID="enc_SomeUsernameToken" >
2951       <!-- Plaintext UsernameToken
2952       <wsse:UsernameToken wsu:Id="SomeUsernameToken" >
2953         ...
2954       </wsse:UsernameToken>
2955       -->
2956     </xenc:EncryptedData>
2957     <ds:KeyInfo>
2958       <wsse:SecurityTokenReference>
2959         <wsse:Reference URI="#_SharedSecretToken" />
2960       </wsse:SecurityTokenReference>
2961     </ds:KeyInfo>
2962     <xenc:EncryptedData>
2963     <wsse:BinarySecurityToken wsu:Id="SomeSupportingToken" >
2964       ...
2965     </wsse:BinarySecurityToken>
2966     <xenc:EncryptedData ID="enc_Signature">
2967       <!-- Plaintext Signature
2968       <ds:Signature Id="Signature">
2969         <ds:SignedInfo>
2970           <ds:References>
2971             <ds:Reference URI="#Timestamp" >...</ds:Reference>
2972             <ds:Reference URI="#SomeUsernameToken" >...</ds:Reference>
2973             <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
2974             <ds:Reference URI="#_SharedSecretToken" >...</ds:Reference>
2975             <ds:Reference URI="#Header1" >...</ds:Reference>
2976             <ds:Reference URI="#Header2" >...</ds:Reference>
2977             <ds:Reference URI="#Body" >...</ds:Reference>
2978           </ds:References>
2979         </ds:SignedInfo>
2980         <ds:SignatureValue>...</ds:SignatureValue>
2981       </ds:Signature>
2982     </xenc:EncryptedData>
2983     <ds:KeyInfo>
2984       <wsse:SecurityTokenReference>
2985         <wsse:Reference URI="#_SharedSecretToken" />
2986       </wsse:SecurityTokenReference>
2987     </ds:KeyInfo>
  </ds:Signature>

```

```

2988 -->
2989 ...
2990 <ds:KeyInfo>
2991   <wsse:SecurityTokenReference>
2992     <wsse:Reference URI="#_SharedSecretToken" />
2993   </wsse:SecurityTokenReference>
2994 </ds:KeyInfo>
2995 </xenc:EncryptedData>
2996 <ds:Signature>
2997   <ds:SignedInfo>
2998     <ds:References>
2999       <ds:Reference URI="#Signature" >...</ds:Reference>
3000       <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
3001     </ds:References>
3002   </ds:SignedInfo>
3003   <ds:SignatureValue>...</ds:SignatureValue>
3004 </ds:Signature>
3005   <ds:KeyInfo>
3006     <wsse:SecurityTokenReference>
3007       <wsse:Reference URI="#SomeSupportingToken" />
3008     </wsse:SecurityTokenReference>
3009   </ds:KeyInfo>
3010 </ds:Signature>
3011 <xenc:ReferenceList>
3012   <xenc:DataReference URI="#enc_Body" />
3013   <xenc:DataReference URI="#enc_Header2" />
3014   ...
3015 </xenc:ReferenceList>
3016 </wsse:Security>
3017 </S:Header>
3018 <S:Body>
3019   <xenc:EncryptedData Id="enc_Body">
3020     ...
3021     <ds:KeyInfo>
3022       <wsse:SecurityTokenReference>
3023         <wsse:Reference URI="#_SharedSecretToken" />
3024       </wsse:SecurityTokenReference>
3025     </ds:KeyInfo>
3026   </xenc:EncryptedData>
3027 </S:Body>
</S:Envelope>

```

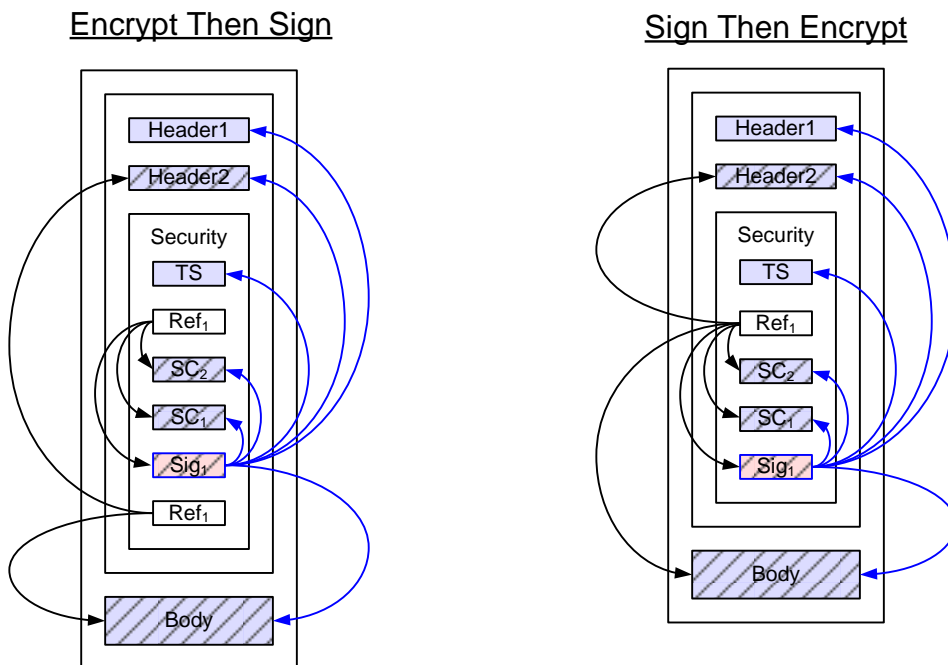
3028 C.2.3 Recipient to Initiator Messages

3029 Messages send from recipient to initiator have the following layout for the security header:

- 3030 1. A `wsu:Timestamp` element if [Timestamp] is 'true'.
- 3031 2. If the `sp:IncludeToken` attribute on the [Encryption Token] is
3032 `.../IncludeToken/Always`, then the [Encryption Token].
- 3033 3. If [Derived Keys] is 'true', then a Derived Key Token, based on the [Encryption
3034 Token]. This Derived Key Token is used for encryption.
- 3035 4. A reference list including references to encrypted items. If [Signature Protection] is
3036 'true', then the reference list MUST include a reference to the message signature
3037 from 6 below, and the `wssell:SignatureConfirmation` elements from 5 below if
3038 any. If [Protection Order] is 'SignBeforeEncrypting', then the reference list MUST
3039 include a reference to all the message parts specified in the EncryptedParts
3040 assertions in the policy. If [Derived Keys] is 'true', then the key in the token from 2
3041 above MUST be used, otherwise the key in the [Encryption Token] from 2 above.
- 3042 5. If [Signature Confirmation] is 'true' then a `wssell:SignatureConfirmation` element
3043 for each signature in the corresponding message sent from initiator to recipient. If

- 3044 there are no signatures in the corresponding message from the initiator to the
 3045 recipient, then a `wssell:SignatureConfirmation` element with no Value attribute.
- 3046 6. If the [Signature Token] is not the same as the [Encryption Token], and the
 3047 `sp:IncludeToken` attribute on the [Signature Token] is `.../IncludeToken/Always`,
 3048 then the [Signature Token].
- 3049 7. If [Derived Keys] is 'true', then a Derived Key Token, based on the [Signature
 3050 Token]. This Derived Key Token is used for signature.
- 3051 8. A signature over the `wsu:Timestamp` from 1 above, any
 3052 `wssell:SignatureConfirmation` elements from 5 above, and all the message parts
 3053 specified in SignedParts assertions in the policy. If [Token Protection] is 'true', the
 3054 signature MUST also cover the [Signature Token] regardless of whether it is included
 3055 in the message. If [Derived Keys] is 'true', the key in the token from 6 above MUST
 3056 be used, otherwise the key in the [Signature Token].
- 3057 9. If [Protection Order] is 'EncryptBeforeSigning' then a reference list referencing all the
 3058 message parts specified in EncryptedParts assertions in the policy. If [Derived Keys]
 3059 is 'true', then the key in the Derived Key Token from 3 above MUST be used,
 3060 otherwise the key in the [Encryption Token].

3061 The following diagram illustrates the security header layout for the recipient to initiator
 3062 message:



3063

3064 The blue arrows (right) indicate parts that were signed as part of the message signature
 3065 labeled `Sig1`. The black arrows (left) from boxes labeled `Ref1` indicate references to parts
 3066 encrypted using a key based on the [SharedSecret Token] (not shown in these diagrams as
 3067 it is referenced as an external token). Two `wssell:SignatureConfirmation` elements
 3068 labeled `SC1` and `SC2` corresponding to the two signatures in the initial message illustrated
 3069 previously is included. In general, the ordering of the items in the security header follows
 3070 the most optimal layout for a receiver to process its contents. The rules used to determine
 3071 this ordering are described in Appendix C.

3072 *Example:*

3073 Recipient to initiator message using EncryptBeforeSigning.

```

3074 <S:Envelope>
3075   <S:Header>
3076     <x:Header1 wsu:Id="Header1" >
3077       ...
3078     </x:Header1>
3079     <wsse1:EncryptedHeader wsu:Id="enc_Header2">
3080       <!-- Plaintext Header2
3081       <x:Header2 wsu:Id="Header2" >
3082         ...
3083       </x:Header2>
3084       -->
3085     </wsse1:EncryptedHeader>
3086     ...
3087   <wsse:Security>
3088     <wsu:Timestamp wsu:Id="Timestamp">
3089       <wsu:Created>...</wsu:Created>
3090       <wsu:Expires>...</wsu:Expires>
3091     </wsu:Timestamp>
3092     <xenc:ReferenceList>
3093       <xenc:DataReference URI="#enc_Signature" />
3094       <xenc:DataReference URI="#enc_SigConf1" />
3095       <xenc:DataReference URI="#enc_SigConf2" />
3096       ...
3097     </xenc:ReferenceList>
3098     <xenc:EncryptedData ID="enc_SigConf1" >
3099       <!-- Plaintext SignatureConfirmation
3100       <wsse1:SignatureConfirmation wsu:Id="SigConf1" >
3101         ...
3102       </wsse1:SignatureConfirmation>
3103       -->
3104     </xenc:EncryptedData>
3105     <xenc:EncryptedData ID="enc_SigConf2" >
3106       <!-- Plaintext SignatureConfirmation
3107       <wsse1:SignatureConfirmation wsu:Id="SigConf2" >
3108         ...
3109       </wsse1:SignatureConfirmation>
3110       -->
3111     </xenc:EncryptedData>
3112     <xenc:EncryptedData Id="enc_Signature">
3113       <!-- Plaintext Signature
3114       <ds:Signature Id="Signature">
3115         <ds:SignedInfo>
3116           <ds:References>
3117             <ds:Reference URI="#Timestamp" >...</ds:Reference>
3118             <ds:Reference URI="#SigConf1" >...</ds:Reference>
3119             <ds:Reference URI="#SigConf2" >...</ds:Reference>
3120             <ds:Reference URI="#Header1" >...</ds:Reference>
3121             <ds:Reference URI="#Header2" >...</ds:Reference>
3122             <ds:Reference URI="#Body" >...</ds:Reference>
3123           </ds:References>
3124         </ds:SignedInfo>
3125         <ds:SignatureValue>...</ds:SignatureValue>
3126         <ds:KeyInfo>
3127           <wsse:SecurityTokenReference>
3128             <wsse:Reference URI="#_SomeIssuedToken" />
3129           </wsse:SecurityTokenReference>
3130         </ds:KeyInfo>
3131       </ds:Signature>
3132       -->
3133     </xenc:EncryptedData>
3134     ...
3135   </wsse:Security>
3136   ...
3137 </S:Envelope>

```

```

3138     <wsse:SecurityTokenReference>
3139         <wsse:Reference URI="#_SomeIssuedToken" />
3140     </wsse:SecurityTokenReference>
3141 </ds:KeyInfo>
3142 <xenc:EncryptedData>
3143 <xenc:ReferenceList>
3144     <xenc:DataReference URI="#enc_Body" />
3145     <xenc:DataReference URI="#enc_Header2" />
3146     ...
3147 </xenc:ReferenceList>
3148 </wsse:Security>
3149 </S:Header>
3150 <S:Body>
3151 <xenc:EncryptedData Id="enc_Body">
3152     ...
3153 <ds:KeyInfo>
3154     <wsse:SecurityTokenReference>
3155         <wsse:Reference URI="#_SomeIssuedToken" />
3156     </wsse:SecurityTokenReference>
3157 </ds:KeyInfo>
3158 </xenc:EncryptedData>
3159 </S:Body>
3160 </S:Envelope>

```

3161 **C.3 Asymmetric Binding**

3162 This section describes how the 'Strict' security header layout rules apply to the Asymmetric
3163 Binding.

3164 **C.3.1 Policy**

3165 The following example shows a policy indicating an Asymmetric Binding, an X509 token as
3166 the [Initiator Token], an X509 token as the [Recipient Token], an algorithm suite, a
3167 requirement to encrypt the message parts before signing, a requirement to encrypt the
3168 message signature, a requirement to include tokens in the message signature and the
3169 supporting signatures, a requirement to include `wsse11:SignatureConfirmation` elements,
3170 a username token attached to the message, and finally an X509 token attached to the
3171 message and endorsing the message signature. Minimum message protection requirements
3172 are described as well.

```

3173 <!-- Example Endpoint Policy -->
3174 <wsp:Policy>
3175   <sp:AsymmetricBinding>
3176     <wsp:Policy>
3177       <sp:RecipientToken>
3178         <wsp:Policy>
3179           <sp:X509Token sp:IncludeToken="../../../IncludeToken/Always" />
3180         </wsp:Policy>
3181       </sp:RecipientToken>
3182       <sp:InitiatorToken>
3183         <wsp:Policy>
3184           <sp:X509Token sp:IncludeToken="../../../IncludeToken/Always" />
3185         </wsp:Policy>
3186       </sp:InitiatorToken>
3187       <sp:AlgorithmSuite>
3188         <wsp:Policy>
3189           <sp:Basic256 />
3190         </wsp:Policy>
3191       </sp:AlgorithmSuite>
3192       <sp:Layout>
3193         <wsp:Policy>
3194           <sp:Strict />
3195         </wsp:Policy>
3196       </sp:Layout>
3197       <sp:IncludeTimestamp />
3198       <sp:EncryptBeforeSigning />
3199       <sp:EncryptSignature />
3200       <sp:ProtectTokens />
3201     </wsp:Policy>
3202   </sp:AsymmetricBinding>
3203   <sp:SignedSupportingTokens>
3204     <wsp:Policy>
3205       <sp:UsernameToken sp:IncludeToken="../../../IncludeToken/Once" />
3206     </wsp:Policy>
3207   </sp:SignedSupportingTokens>
3208   <sp:SignedEndorsingSupportingTokens>
3209     <wsp:Policy>
3210       <sp:X509V3Token sp:IncludeToken="../../../IncludeToken/Once" />
3211     </wsp:Policy>
3212   </sp:SignedEndorsingSupportingTokens>
3213   <sp:Wss11>
3214     <wsp:Policy>
3215       <sp:RequireSignatureConfirmation />
3216     </wsp:Policy>
3217   </sp:Wss11>
3218 </wsp:Policy>
3219
3220 <!-- Example Message Policy -->
3221 <wsp>All>
3222   <sp:SignedParts>
3223     <sp:Header Name="Header1" Namespace="../../../" />
3224     <sp:Header Name="Header2" Namespace="../../../" />
3225   </sp:SignedParts>
3226   <sp:EncryptedParts>
3227     <sp:Header Name="Header2" Namespace="../../../" />
3228   </sp:EncryptedParts>
3229 </wsp>All>
3230
3231

```

3232
3233 This policy is used as the basis for the examples shown in the subsequent section describing
3234 the security header layout for this binding.

3235 C.3.2 Initiator to Recipient Messages

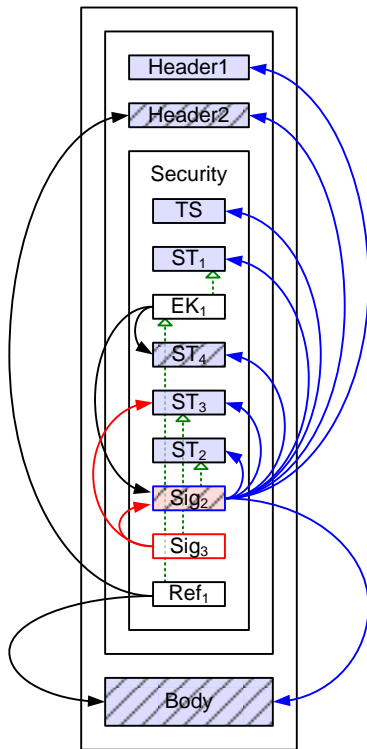
3236 Messages sent from initiator to recipient have the following layout:

- 3237 1. A `wsu:Timestamp` element if [Timestamp] is 'true'.
- 3238 2. If a [Recipient Token] is specified, and the associated `sp:IncludeToken` attribute is
3239 `.../IncludeToken/Once` or `.../IncludeToken/Always`, then the [Recipient Token].
- 3240 3. If a [Recipient Token] is specified and [Protection Order] is 'SignBeforeEncrypting' or
3241 [SignatureProtection] is 'true' then an `xenc:EncryptedKey` element, containing a key
3242 encrypted for the recipient. The `xenc:EncryptedKey` element MUST include an
3243 `xenc:ReferenceList` containing a reference to all the message parts specified in
3244 EncryptedParts assertions in the policy. If [Signature Protection] is 'true' then the
3245 reference list MUST contain a reference to the message signature from 6 below. It is
3246 an error if [Signature Protection] is 'true' and there is not a message signature.
- 3247 4. Any tokens from the [Signed Supporting Tokens] and [Signed Endorsing Supporting
3248 Tokens] properties whose `sp:IncludeToken` attribute is `.../IncludeToken/Once` or
3249 `.../IncludeToken/Always`.
- 3250 5. If an [Initiator Token] is specified, and the associated `sp:IncludeToken` attribute is
3251 `.../IncludeToken/Once` or `.../IncludeToken/Always`, then the [Initiator Token].
- 3252 6. A signature based on the key in the [Initiator Token] if specified, over the
3253 `wsu:Timestamp` from 1 above, any tokens from 4 above regardless of whether they
3254 are included in the message, and any message parts specified in SignedParts
3255 assertions in the policy. If [Token Protection] is 'true', the signature MUST also cover
3256 the [Initiator Token] regardless of whether it is included in the message.
- 3257 7. Signatures for tokens from the [Endorsing Supporting Tokens] and [Signed
3258 Endorsing Supporting Tokens] properties. If [Derived Keys] is 'true' and the
3259 supporting token is associated with a symmetric key, then a Derived Key Token,
3260 based on the supporting token, appears before the signature. If [Token Protection] is
3261 'true', the signature MUST also cover the supporting token regardless of whether it is
3262 included in the message.
- 3263 8. If a [Recipient Token] is specified and [Protection Order] is 'EncryptBeforeSigning'
3264 then if [Signature Protection] is 'false' then an `xenc:EncryptedKey` element,
3265 containing a key encrypted for the recipient and a reference list, else if [Signature
3266 Protection] is 'true', a reference list. The reference list includes a reference to all the
3267 message parts specified in EncryptedParts assertions in the policy. The encrypted
3268 parts MUST reference the key contained in the `xenc:EncryptedKey` element from 3
3269 above.

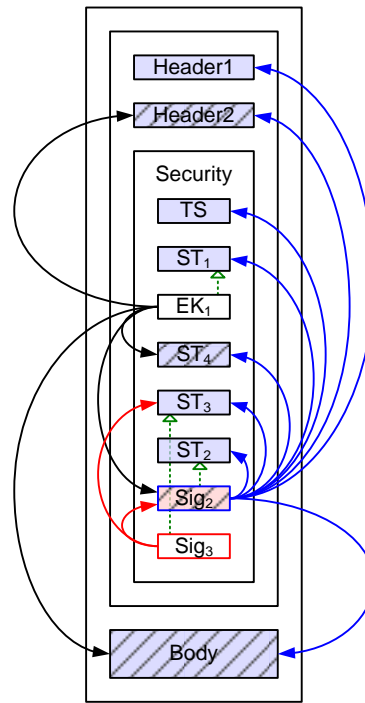
3270

3271 The following diagram illustrates the security header layout for the initiator to recipient
3272 messages:

Encrypt Then Sign



Sign Then Encrypt



3273

3274 The blue arrows (right) indicate parts that were signed as part of the message signature
3275 labeled Sig₂ using the [Initiator Token] labeled ST₂. The red arrows (left) from the box
3276 labeled Sig₃ indicate the parts signed by the supporting token ST₃, namely the message
3277 signature Sig₂ and the token used as the basis for the signature labeled ST₃. The black
3278 arrows (left) from boxes labeled EK₁ indicate references to parts encrypted using a key
3279 encrypted for the [Recipient Token] labeled ST₁. The black arrows (left) from boxes labeled
3280 Ref₁ indicate additional references to parts encrypted using the key contained in the
3281 encrypted key labeled EK₁. The green dotted arrows indicate the token used as the basis for
3282 each cryptographic operation. In general, the ordering of the items in the security header
3283 follows the most optimal layout for a receiver to process its contents. The rules used to
3284 determine this ordering are described in Appendix C.

3285

3286 Note: In most typical scenarios, the recipient key is not included in the message, but rather
3287 the encrypted key contains an external reference to the token containing the encryption
3288 key. The diagram illustrates how one might attach a security token related to the encrypted
3289 key for completeness. One possible use-case for this approach might be a stack which does
3290 not support the STR Dereferencing Transform, but wishes to include the encryption token in
3291 the message signature.

3292 Initiator to recipient message *Example*

3293

```

3294 <S:Envelope>
3295   <S:Header>
3296     <x:Header1 wsu:Id="Header1" >
3297       ...
3298     </x:Header1>
3299     <wsse1:EncryptedHeader wsu:Id="enc_Header2">
3300       <!-- Plaintext Header2
3301       <x:Header2 wsu:Id="Header2" >
3302         ...
3303       </x:Header2>
3304       -->
3305     </wsse1:EncryptedHeader>
3306     ...
3307   <wsse:Security>
3308     <wsu:Timestamp wsu:Id="Timestamp">
3309       <wsu:Created>...</wsu:Created>
3310       <wsu:Expires>...</wsu:Expires>
3311     </wsu:Timestamp>
3312     <wsse:BinarySecurityToken wsu:Id="RecipientToken" >
3313       ...
3314     </wsse:BinarySecurityToken>
3315     <xenc:EncryptedKey wsu:Id="RecipientEncryptedKey" >
3316       ...
3317     <xenc:ReferenceList>
3318       <xenc:DataReference URI="#enc_Signature" />
3319       <xenc:DataReference URI="#enc_SomeUsernameToken" />
3320       ...
3321     </xenc:ReferenceList>
3322   </xenc:EncryptedKey>
3323   <xenc:EncryptedData ID="enc_SomeUsernameToken" >
3324     <!-- Plaintext UsernameToken
3325     <wsse:UsernameToken wsu:Id="SomeUsernameToken" >
3326       ...
3327     </wsse:UsernameToken>
3328     -->
3329     ...
3330   </xenc:EncryptedData>
3331   <wsse:BinarySecurityToken wsu:Id="SomeSupportingToken" >
3332     ...
3333   </wsse:BinarySecurityToken>
3334   <wsse:BinarySecurityToken wsu:Id="InitiatorToken" >
3335     ...
3336   </wsse:BinarySecurityToken>
3337   <xenc:EncryptedData ID="enc_Signature">
3338     <!-- Plaintext Signature
3339     <ds:Signature Id="Signature">
3340       <ds:SignedInfo>
3341         <ds:References>
3342           <ds:Reference URI="#Timestamp" >...</ds:Reference>
3343           <ds:Reference URI="#SomeUsernameToken" >...</ds:Reference>
3344           <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
3345           <ds:Reference URI="#RecipientToken" >...</ds:Reference>
3346           <ds:Reference URI="#InitiatorToken" >...</ds:Reference>
3347           <ds:Reference URI="#Header1" >...</ds:Reference>
3348           <ds:Reference URI="#Header2" >...</ds:Reference>
3349           <ds:Reference URI="#Body" >...</ds:Reference>
3350         </ds:References>
3351       </ds:SignedInfo>
3352     </ds:SignatureValue>...</ds:SignatureValue>
3353     <ds:KeyInfo>
3354       <wsse:SecurityTokenReference>
3355       <wsse:Reference URI="#InitiatorToken" />
3356     </wsse:SecurityTokenReference>
3357

```

```

3358     </ds:KeyInfo>
3359     </ds:Signature>
3360     -->
3361     ...
3362 </xenc:EncryptedData>
3363 <ds:Signature>
3364   <ds:SignedInfo>
3365     <ds:References>
3366       <ds:Reference URI="#Signature" >...</ds:Reference>
3367       <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
3368     </ds:References>
3369   </ds:SignedInfo>
3370   <ds:SignatureValue>...</ds:SignatureValue>
3371   <ds:KeyInfo>
3372     <wsse:SecurityTokenReference>
3373       <wsse:Reference URI="#SomeSupportingToken" />
3374     </wsse:SecurityTokenReference>
3375   </ds:KeyInfo>
3376 </ds:Signature>
3377 <xenc:ReferenceList>
3378   <xenc:DataReference URI="#enc_Body" />
3379   <xenc:DataReference URI="#enc_Header2" />
3380   ...
3381 </xenc:ReferenceList>
3382 </wsse:Security>
3383 </S:Header>
3384 <S:Body>
3385   <xenc:EncryptedData Id="enc_Body">
3386     ...
3387     <ds:KeyInfo>
3388       <wsse:SecurityTokenReference>
3389         <wsse:Reference URI="#RecipientEncryptedKey" />
3390       </wsse:SecurityTokenReference>
3391     </ds:KeyInfo>
3392   </xenc:EncryptedData>
3393 </S:Body>
3394 </S:Envelope>

```

3395 C.3.3 Recipient to Initiator Messages

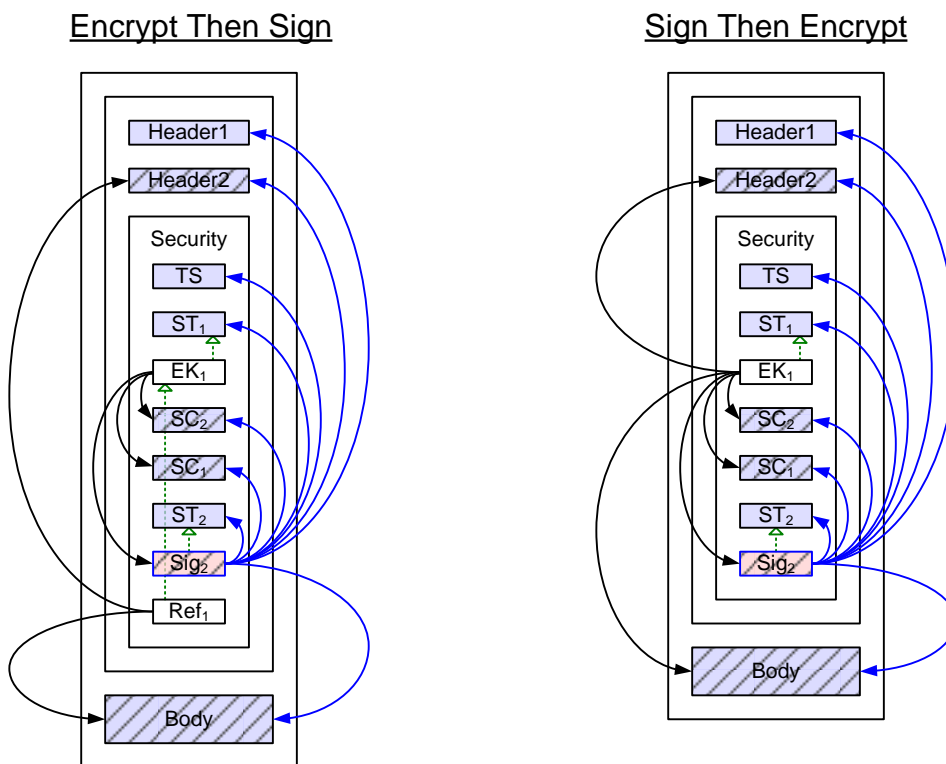
3396 Messages sent from recipient to initiator have the following layout:

- 3397 1. A `wsu:Timestamp` element if [Timestamp] is 'true'.
- 3398 2. If an [Initiator Token] is specified, and the associated `sp:IncludeToken` attribute is
3399 `.../IncludeToken/Always`, then the [Initiator Token].
- 3400 3. If an [Initiator Token] is specified and [Protection Order] is 'SignBeforeEncrypting' or
3401 [SignatureProtection] is 'true' then an `xenc:EncryptedKey` element, containing a key
3402 encrypted for the initiator. The `xenc:EncryptedKey` element MUST include an
3403 `xenc:ReferenceList` containing a reference to all the message parts specified in
3404 EncryptedParts assertions in the policy. If [Signature Protection] is 'true' then the
3405 reference list MUST also contain a reference to the message signature from 6 below,
3406 if any and references to the `wss11:SignatureConfirmation` elements from 4
3407 below, if any.
- 3408 4. If [Signature Confirmation] is 'true', then a `wss11:SignatureConfirmation`
3409 element for each signature in the corresponding message sent from initiator to
3410 recipient. If there are no signatures in the corresponding message from the initiator
3411 to the recipient, then a `wss11:SignatureConfirmation` element with no Value
3412 attribute.

- 3413 5. If a [Recipient Token] is specified, and the associated `sp:IncludeToken` attribute is
 3414 `.../IncludeToken/Always`, then the [Recipient Token].
- 3415 6. If a [Recipient Token] is specified, then a signature based on the key in the
 3416 [Recipient Token], over the `wsu:Timestamp` from 1 above, the
 3417 `wsse11:SignatureConfirmation` elements from 4 above, and any message parts
 3418 specified in SignedParts assertions in the policy. If [Token Protection] is 'true' and
 3419 the [Initiator Token] is specified, then the signature MUST also cover the [Initiator
 3420 Token].
- 3421 7. If an [Initiator Token] is specified and [Protection Order] is 'EncryptBeforeSigning'
 3422 then if [Signature Protection] is 'false' then an `xenc:EncryptedKey` element,
 3423 containing a key encrypted for the recipient and a reference list, else if [Signature
 3424 Protection] is 'true', a reference list. The reference list includes a reference to all the
 3425 message parts specified in EncryptedParts assertions in the policy. The encrypted
 3426 parts MUST reference the key contained in the `xenc:EncryptedKey` element from 3
 3427 above.

3428

3429 The following diagram illustrates the security header layout for the recipient to initiator
 3430 messages:



3431

3432 The blue arrows (right) indicate parts that were signed as part of the message signature
 3433 labeled Sig₂ using the [Recipient Token] labeled ST₂. The black arrows (left) from boxes
 3434 labeled EK₁ indicate references to parts encrypted using a key encrypted for the [Recipient
 3435 Token] labeled ST₁. The black arrows (left) from boxes labeled Ref₁ indicate additional
 3436 references to parts encrypted using the key contained in the encrypted key labeled EK₁. The
 3437 green dotted arrows indicate the token used as the basis for each cryptographic operation.
 3438 Two `wsse11:SignatureConfirmation` elements labeled SC₁ and SC₂ corresponding to the
 3439 two signatures in the initial message illustrated previously is included. In general, the

- 3440 ordering of the items in the security header follows the most optimal layout for a receiver to
- 3441 process its contents. The rules used to determine this ordering are described in Appendix C.
- 3442 Recipient to initiator message *Example:*

```

3443 <S:Envelope>
3444   <S:Header>
3445     <x:Header1 wsu:Id="Header1" >
3446       ...
3447     </x:Header1>
3448     <wssell:EncryptedHeader wsu:Id="enc_Header2">
3449       <!-- Plaintext Header2
3450       <x:Header2 wsu:Id="Header2" >
3451         ...
3452       </x:Header2>
3453       -->
3454       ...
3455     </wssell:EncryptedHeader>
3456     ...
3457     <wsse:Security>
3458       <wsu:Timestamp wsu:Id="Timestamp">
3459         <wsu:Created>...</wsu:Created>
3460         <wsu:Expires>...</wsu:Expires>
3461       </wsu:Timestamp>
3462       <wsse:BinarySecurityToken wsu:Id="InitiatorToken" >
3463         ...
3464       </wsse:BinarySecurityToken>
3465       <xenc:EncryptedKey wsu:Id="InitiatorEncryptedKey" >
3466         ...
3467         <xenc:ReferenceList>
3468           <xenc:DataReference URI="#enc_Signature" />
3469           <xenc:DataReference URI="#enc_SigConf1" />
3470           <xenc:DataReference URI="#enc_SigConf2" />
3471           ...
3472         </xenc:ReferenceList>
3473       </xenc:EncryptedKey>
3474       <xenc:EncryptedData ID="enc_SigConf2" >
3475         <!-- Plaintext SignatureConfirmation
3476         <wssell:SignatureConfirmation wsu:Id="SigConf2" ...>
3477           ...
3478         </wssell:SignatureConfirmation>
3479         -->
3480         ...
3481       </xenc:EncryptedData>
3482       <xenc:EncryptedData ID="enc_SigConf1" >
3483         <!-- Plaintext SignatureConfirmation
3484         <wssell:SignatureConfirmation wsu:Id="SigConf1" ...>
3485           ...
3486         </wssell:SignatureConfirmation>
3487         -->
3488         ...
3489       </xenc:EncryptedData>
3490       <wsse:BinarySecurityToken wsu:Id="RecipientToken" >
3491         ...
3492       </wsse:BinarySecurityToken>
3493       <xenc:EncryptedData ID="enc_Signature">
3494         <!-- Plaintext Signature
3495         <ds:Signature Id="Signature">
3496           <ds:SignedInfo>
3497             <ds:References>
3498               <ds:Reference URI="#Timestamp" >...</ds:Reference>
3499               <ds:Reference URI="#SigConf1" >...</ds:Reference>
3500               <ds:Reference URI="#SigConf2" >...</ds:Reference>
3501               <ds:Reference URI="#RecipientToken" >...</ds:Reference>
3502               <ds:Reference URI="#InitiatorToken" >...</ds:Reference>
3503               <ds:Reference URI="#Header1" >...</ds:Reference>
3504               <ds:Reference URI="#Header2" >...</ds:Reference>
3505               <ds:Reference URI="#Body" >...</ds:Reference>
3506             </ds:References>

```

```

3507     </ds:SignedInfo>
3508     <ds:SignatureValue>...</ds:SignatureValue>
3509     <ds:KeyInfo>
3510         <wsse:SecurityTokenReference>
3511             <wsse:Reference URI="#RecipientToken" />
3512         </wsse:SecurityTokenReference>
3513     </ds:KeyInfo>
3514 </ds:Signature>
3515     →
3516     ...
3517 </xenc:EncryptedData>
3518 <xenc:ReferenceList>
3519     <xenc:DataReference URI="#enc_B"dy" />
3520     <xenc:DataReference URI="#enc_Head"r2" />
3521     ...
3522 </xenc:ReferenceList>
3523 </wsse:Security>
3524 </S:Header>
3525 <S:Body>
3526     <xenc:EncryptedData "d="enc_B"dy">
3527         ...
3528         <ds:KeyInfo>
3529             <wsse:SecurityTokenReference>
3530                 <wsse:Reference URI="#InitiatorEncrypted"ey" />
3531             </wsse:SecurityTokenReference>
3532         </ds:KeyInfo>
3533     </xenc:EncryptedData>
3534 </S:Body>
3535 </S:Envelope>

```

3536
3537

D.Signed and Encrypted Elements in the Security Header

3538
3539
3540
3541
3542
3543

This section lists the criteria for when various child elements of the Security header are signed and/or encrypted at the message level including whether they are signed by the message signature or a supporting signature. It assumes that there are no `sp:SignedElements` and no `sp:EncryptedElements` assertions in the policy. If such assertions are present in the policy then additional child elements of the security header might be signed and/or encrypted.

3544

D.1 Elements signed by the message signature

3545
3546
3547
3548
3549
3550
3551

1. The `wsu:Timestamp` element.
2. All `wssell:SignatureConfirmation` elements.
3. Security Tokens corresponding to [Initiator Signature Token],[Recipient Signature Token], [Initiator Encryption Token], [Recipient Encryption Token], [Signature Token] or [Encryption Token] when [Token Protection] has a value of 'true'.
4. Security Tokens corresponding to [Signed Supporting Tokens] or [Signed Endorsing Supporting Tokens].

3552

D.2 Elements signed by all endorsing signatures

3553
3554

1. The `ds:Signature` element that forms the message signature.
2. The `wsu:Timestamp` element in the case of a transport binding.

3555

D.3 Elements signed by a specific endorsing signature

3556
3557

1. Security Tokens corresponding to [Endorsing Supporting Tokens] or [Signed Endorsing Supporting Tokens] when [Token Protection] has a value of 'true'

3558

D.4 Elements that are encrypted

3559
3560
3561
3562
3563

1. The `ds:Signature` element that forms the message signature when [Signature Protection] has a value of 'true'.
2. All `wssell:SignatureConfirmation` elements when [Signature Protection] has a value of 'true'.
3. Any `wsse:UsernameToken` when a transport binding is NOT being used.

3564 **E. Acknowledgements**

3565 The following individuals have participated in the creation of this specification and are
3566 gratefully acknowledged:

3567 **Original Authors:**

3568 Giovanni Della-Libera, Microsoft

3569 Martin Gudgin, Microsoft

3570 Phillip Hallam-Baker, VeriSign

3571 Maryann Hondo, IBM

3572 Hans Granqvist, Verisign

3573 Chris Kaler, Microsoft (editor)

3574 Hiroshi Maruyama, IBM

3575 Michael McIntosh, IBM

3576 Anthony Nadalin, IBM (editor)

3577 Nataraj Nagaratnam, IBM

3578 Rob Philpott, RSA Security

3579 Hemma Prafullchandra, VeriSign

3580 John Shewchuk, Microsoft

3581 Doug Walter, Microsoft

3582 Riaz Zolfonoon, RSA Security

3583

3584 **Original Acknowledgements:**

3585 Vaithialingam B. Balayoghan, Microsoft

3586 Francisco Curbera, IBM

3587 Christopher Ferris, IBM

3588 Cédric Fournet, Microsoft

3589 Andy Gordon, Microsoft

3590 Tomasz Janczuk, Microsoft

3591 David Melgar, IBM

3592 Mike Perks, IBM

3593 Bruce Rich, IBM

3594 Jeffrey Schlimmer, Microsoft

3595 Chris Sharp, IBM

3596 Kent Tamura, IBM

3597 T.R. Vishwanath, Microsoft

3598 Elliot Waingold, Microsoft

F. Non-Normative Text

3600

G.Revision History

3601 [optional; should not be included in OASIS Standards]

3602

Revision	Date	Editor	Changes Made
0.1	12-06-2005	Anthony Nadalin	Initial Conversion to OASIS Template
0.2	01-09-2006	Martin Gudgin	Updated TOC. Typos. Namespaces.
0.3.	01-17-2006	Marc Goodner	Updated artifact identifier per AIR guidelines, corrected version number, 2005 updated to 2006, added resolution of i012, changed status to editor draft
0.4	02-20-2006	Marc Goodner	Corrected section numbers after section reordering caused by OASIS template (Issue 21)
0.5	03-27-2006	Martin Gudgin	Issue 3 - Lines 229-236 Issue 9 - Lines 1620-1644, 1675, 1681, 1685-1708 Issue 23 - Lines 1300 and 1302 Issue 25 - Lines 1921-1932 Issue 26 - Line 1343 Issue 27 - Lines 766-783 Issue 29 - 1704-1707 Issue 32 - Line 828 (Text was removed not added) Issue 49 - Lines 1273, 1277-1278 Issue 50 - Lines 557-560
0.6	04-27-2006	Martin Gudgin	Issue 16 – Section 4.1.1 Issue 18 – Sections 1.6, 6.1, 7.1 Issue 30 – Section 11 Issue 51 – Sections 5.2.1, 5.2.2, 5.2.3, 5.3.1, 5.3.2, 5.3.3, 5.3.4, 5.3.5, 5.3.6, 5.3.7, 5.3.8, 5.3.9 Issue 53 – Section 5.3.7
0.7	06-19-2006	Martin Gudgin	Issue 31 – Updates to Section 5.3.1 Issue 33 – Added new Appendix D Issue 48 – Updates to Sections 7.4, 7.5 and Appendix A. Issue 69 – Updates to Sections 5.3 and 7.2. Issue 75 – Updates to Section 5.3.10

3603
3604