



WS-SecurityPolicy v1.0

Working Draft, 08 December 2005

Artifact Identifier:

ws-sx-spec-draft-v1-r0-ws-securitypolicy

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 [TC name | membership of OASIS] 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 2005. 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
1.5	Terminology and Notation.....	Error! Bookmark not defined.
1.5.1	Notational Conventions.....	8
1.6	Normative References.....	8
1.7	Non-Normative References.....	9
2	Security Policy Model.....	10
2.1	Security Assertion Model.....	10
2.2	Nested Policy Assertions.....	11
2.3	Security Binding Abstraction.....	11
3	Policy Considerations.....	13
3.1	Nested Policy.....	13
3.1.1	Nesting Policy Elements.....	13
3.1.2	Nested Policy Assertions.....	14
3.1.3	Nesting Policy Processing Rules.....	15
3.1.4	Nested Policy Normalization Worked Example.....	15
3.1.5	Nested Policy Intersection Worked Example.....	16
3.2	Policy Subjects.....	18
4	Protection Assertions.....	20
4.1	Integrity Assertions.....	20
4.1.1	SignedParts Assertion.....	20
4.1.2	SignedElements Assertion.....	21
4.2	Confidentiality Assertions.....	21
4.2.1	EncryptedParts Assertion.....	22
4.2.2	EncryptedElements Assertion.....	23
4.3	Required Elements Assertion.....	23
4.3.1	RequiredElements Assertion.....	23
5	Token Assertions.....	25
5.1	Token Inclusion.....	25
5.1.1	Token Inclusion Values.....	25
5.2	Token Properties.....	26
5.2.1	[Derived Keys] Property.....	26
5.3	Token Assertions.....	26
5.3.1	UsernameToken Assertion.....	26
5.3.2	IssuedToken Assertion.....	27
5.3.3	X509Token Assertion.....	28
5.3.4	KerberosToken Assertion.....	29
5.3.5	SpnegoContextToken Assertion.....	30
5.3.6	SecurityContextToken Assertion.....	31
5.3.7	SecureConversationToken Assertion.....	31

5.3.8	SamlToken Assertion	33
5.3.9	RelToken Assertion	35
5.3.10	HttpsToken Assertion	36
6	Security Binding Properties	37
6.1	[Algorithm Suite] Property	37
6.2	[Timestamp] Property	39
6.3	[Protection Order] Property	39
6.4	[Signature Protection] Property	39
6.5	[Token Protection] Property	40
6.6	[Entire Header and Body Signatures] Property	40
6.7	[Security Header Layout] Property	40
6.7.1	Strict Layout Rules	40
7	Security Binding Assertions.....	42
7.1	AlgorithmSuite Assertion	42
7.2	Layout Assertion	44
7.3	TransportBinding Assertion	44
7.4	SymmetricBinding Assertion.....	45
7.5	AsymmetricBinding Assertion.....	47
8	Supporting Tokens	49
8.1	SupportingTokens Assertion.....	50
8.2	SignedSupportingTokens Assertion	51
8.3	EndorsingSupportingTokens Assertion	53
8.4	SignedEndorsingSupportingTokens Assertion.....	54
8.5	Example	56
9	WSS: SOAP Message Security Options	58
9.1	Wss10 Assertion.....	59
9.2	Wss11 Assertion	60
10	WS-Trust Options	62
10.1	Trust10 Assertion.....	63
11	Security Considerations	64
A.	Assertions and WS-PolicyAttachment.....	65
A.1	Endpoint Policy Subject Assertions.....	65
A.1.1	Security Binding Assertions	65
A.1.2	Token Assertions	65
A.1.3	WSS: SOAP Message Security 1.0 Assertions	65
A.1.4	WSS: SOAP Message Security 1.1 Assertions	65
A.1.5	Trust 1.0 Assertions	65
A.2	Operation Policy Subject Assertions	65
A.2.1	Supporting Token Assertions.....	65
A.3	Message Policy Subject Assertions	65
A.3.1	Supporting Token Assertions.....	65
A.3.2	Protection Assertions	66
A.4	Assertions With Undefined Policy Subject	66
A.4.1	General Assertions	66
A.4.2	Token Usage Assertions.....	66

A.4.3 Token Assertions	66
A.4.4 WSS: SOAP Message Security 1.0 Assertions	67
A.4.5 WSS: SOAP Message Security 1.1 Assertions	67
A.4.6 Trust 1.0 Assertions	67
B. Issued Token Policy	68
C. Strict Security Header Layout Examples.....	70
C.1 Transport Binding	70
C.1.1 Policy	70
C.1.2 Initiator to Recipient Messages.....	71
C.1.3 Recipient to Initiator Messages.....	73
C.2 Symmetric Binding	75
C.2.1 Policy	75
C.2.2 Initiator to Recipient Messages.....	77
C.2.3 Recipient to Initiator Messages.....	80
C.3 Asymmetric Binding	84
C.3.1 Policy	84
C.3.2 Initiator to Recipient Messages.....	86
C.3.3 Recipient to Initiator Messages.....	89
D. Acknowledgements	93
E. Non-Normative Text	94
F. Revision History.....	95

1 Introduction

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

Sections 3.4, 11, 12, all examples and all Appendices are non-normative.

1.1 Example

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

Table 1: Example security policy.

```
(01) <wsp:Policy>
(02)   <sp:SymmetricBinding>
(03)     <wsp:Policy>
(04)       <sp:ProtectionToken>
(05)         <wsp:Policy>
(06)           <sp:KerberosV5APREQToken
(07)             sp:IncludeToken=".../IncludeToken/Once" />
(08)           </sp:KerberosV5APREQToken>
(09)         </wsp:Policy>
(10)       </sp:ProtectionToken>
(11)     <sp:SignBeforeEncrypting />
(12)     <sp:EncryptSignature />
(13)   </wsp:Policy>
(14) </sp:SymmetricBinding>
(15) <sp:SignedParts>
(16)   <sp:Body/>
(17)   <sp:Header
(18)     Namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
(19)   />
(20) </sp:SignedParts>
(21) <sp:EncryptedParts>
(22)   <sp:Body/>
(23) </sp:EncryptedParts>
(24) </wsp:Policy>
```

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

56 1.2 Namespaces

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

58 `http://schemas.xmlsoap.org/ws/2005/07/securitypolicy`

59

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

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

Prefix	Namespace	Specification(s)
S	http://schemas.xmlsoap.org/soap/envelope/	[SOAP11]
ds	http://www.w3.org/2000/09/xmldsig#	[XMLDSIG]
enc	http://www.w3.org/2001/04/xmlenc#	[XMLENC]
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	[WSS10]
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd	[WSS10]
wsse11	http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-wss-wssecurity-secext-1.1.xsd	[WSS11]
wsp	http://schemas.xmlsoap.org/ws/2004/09/policy	[WS-Policy], [WS-PolicyAttachment]
xsd	http://www.w3.org/2001/XMLSchema	[XMLSchema Part1], [XMLSchema Part2]
wst	http://schemas.xmlsoap.org/ws/2005/02/trust	[WS-Trust]
wsc	http://schemas.xmlsoap.org/ws/2005/02/sc	[WS-SecureConversation]
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing	[WS-Addressing]
sp	http://schemas.xmlsoap.org/ws/2005/07/securitypolicy	This specification

63 1.3 Schema Files

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

66 <http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.xsd>

67 1.4 Terminology

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

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

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

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

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

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

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

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

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

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

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

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

84 1.4.1 Notational Conventions

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

88

89 In this document reference is made to the `wsu:Id` attribute and the `wsu:Created` and
90 `wsu:Expires` elements in a utility schema ([http://docs.oasis-open.org/wss/2004/01/oasis-
91 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
92 `wsu:Expires` elements were added to the utility schema with the intent that other
93 specifications requiring such an ID or timestamp could reference it (as is done here).

94

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

101 1.5 Normative References

102 [RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
103 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2119, March 1997.

104 [KEYWORDS] S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," [RFC](#)
105 [2119](#), Harvard University, March 1997
106 [RFC2068] IETF Standard, "[Hypertext Transfer Protocol -- HTTP/1.1](#)" January 1997
107 [SOAP11] W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.
108 [SOAP12] W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging Framework," 24
109 June 2003.
110 [XMLSchema Part1] W3C Recommendation, "[XML Schema Part 1: Structure Second Edition](#)," 28
111 October 2004.
112 [XMLSchema Part2] W3C Recommendation, "[XML Schema Part 2: Datatypes Second Edition](#)," 28
113 October 2004.
114 [URI] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI):
115 Generic Syntax," RFC 3986, MIT/LCS, Day Software, Adobe Systems, January
116 2005.

117 **1.6 Non-Normative References**

118 [WS-Policy] S.Bajaj, et.al., "[Web Services Policy Framework \(WS-Policy\)](#)," September 2004
119 [WS-PolicyAttachment] S.Bajaj, et.al., "[Web Services Policy Attachment \(WS-PolicyAttachment\)](#),"
120 September 2004
121 [WS-Trust] S.Anderson, et.al., "[Web Services Trust Language \(WS-Trust\)](#)," February 2005
122 [WS-SecureConversation] S.Anderson, et.al., "[Web Services Secure Conversation Language \(WS-
123 SecureConversation\)](#)," February 2005
124 [WS-Addressing] D.Box, et.al., "[Web Services Addressing Language \(WS-Addressing\)](#)," August
125 2004
126 [WSS10] A. Nadalin, et.al., "[Web Services Security: SOAP Message Security 1.0 \(WS-
127 Security 2004\)](#)," OASIS Standard 200401, March 2004
128 [WSS:UsernameToken 1.0] A. Nadalin, et.al., "[Web Services Security: UsernameToken Profile 1.0](#),"
129 OASIS Standard 200401, March 2004
130 [WSS:X509Token] P. Hallam-Baker, et.al., "[Web Services Security X.509 Certificate Token Profile](#),"
131 OASIS Standard 200401, March 2004
132 [WSS:Kerberos Token 1.0] TBD – update
133 [XMLDSIG] D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. *XML-
134 Signature Syntax and Processing*, W3C Recommendation, 12 February 2002.
135 <http://www.w3.org/TR/xmlsig-core/>.
136 [XMLENC] W3C Recommendation, "[XML Encryption Syntax and Processing](#)," 10 December
137 2002.

138 2 Security Policy Model

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

144
145 The primary goal of this specification is to define an initial set of patterns or sets of
146 assertions that represent common ways to describe how messages are secured on a
147 communication path. The intent is to allow flexibility in terms of the tokens, cryptography,
148 and mechanisms used, including leveraging transport security, but to be specific enough to
149 ensure interoperability based on assertion matching.

150
151 It is a goal of the security policy model to leverage the WS-Policy framework's intersection
152 algorithm for selecting policy alternatives and the attachment mechanism for associating
153 policy assertions with web service artifacts. Consequently, wherever possible, the security
154 policy assertions do not use parameters or attributes. This enables first-level, QName based
155 assertion matching without security domain-specific knowledge to be done at the framework
156 level. The first level matching is intended to provide a narrowed set of policy alternatives
157 that are shared by the two parties attempting to establish a secure communication path.

158
159 In general, assertions defined in this specification allow additional attributes, based on
160 schemas, to be added on to the assertion element as an extensibility mechanism but the
161 WS-Policy framework will not match based on these attributes. Attributes specified on the
162 assertion element that are not defined in this specification or in WS-Policy are to be treated
163 as informational properties.

164 2.1 Security Assertion Model

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

173
174 To indicate the scope of protection, assertions identify message parts that are to be
175 protected in a specific way, such as integrity or confidentiality protection, and are referred
176 to as protection assertions.

177
178 The general aspects of security includes the relationships between or characteristics of the
179 environment in which security is being applied, such as the tokens being used, which are for

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

182

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

189

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

194 **2.2 Nested Policy Assertions**

195 Assertions may be used to further qualify a specific aspect of another assertion. For
196 example, an assertion describing the set of algorithms to use may qualify the specific
197 behavior of a security binding. To enable this set of functionality, this specification
198 introduces a mechanism for nesting policy assertions underneath other assertions. This
199 mechanism is described in Section 4.

200 **2.3 Security Binding Abstraction**

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

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

208

209 A binding defines the following security characteristics:

- 210 • The minimum set of tokens that will be used and how they are bound to messages
- 211 • Any necessary key transfer mechanisms
- 212 • Any required message elements (e.g. timestamps)
- 213 • The content and ordering of elements in the `wsse:Security` header. Elements not
214 specified in the binding are not allowed.
- 215 • How correlation of messages is performed securely (if applicable to the message
216 pattern)

217

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

221

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

- 226 • TransportBinding
- 227 • SymmetricBinding
- 228 • AsymmetricBinding

229 3 Policy Considerations

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

232 3.1 Nested Policy

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

239

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

241 3.1.1 Nesting Policy Elements

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

245

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

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

```
256 <wsp:Policy>  
257   <A />  
258   <B />  
259 </wsp:Policy>  
260 <wsp:Policy>  
261   <A />  
262   <C />  
263 </wsp:Policy>
```

264

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

```
267 <wsp:Policy>
268   <wsp:All>
269     <A />
270     <wsp:ExactlyOne>
271       <B />
272       <C />
273     </wsp:ExactlyOne>
274   </wsp:All>
275 </wsp:Policy>
```

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

277 3.1.2 Nested Policy Assertions

278 Some assertions may need to declare that additional assertions, scoped only to that
279 assertion, further qualify the behavior and compatibility semantics of that assertion.
280 Whereas the `wsp:All` and `wsp:ExactlyOne` elements describe requirements and alternatives
281 of a `wsp:Policy` element, nested assertions describe requirements and alternatives for the
282 enclosing assertion element. To enable these semantics, this specification defines some
283 assertions such that they have a single `wsp:Policy` child element which in turn contains
284 assertions which qualify the behavior of the enclosing assertion. Two such assertions are
285 compatible if they have the same QName AND their nested policy expressions (if any) are
286 compatible.

287

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

```
290 <wsp:Policy>
291   <A />
292   <B>
293     <wsp:Policy>
294       <C />
295       <D />
296     </wsp:Policy>
297   </B>
298 </wsp:Policy>
```

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

301

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

305

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

```
312 <wsp:Policy>
313   <A />
314   <B x:ContainsPolicy="true">
315     <wsp:Policy>
316       ...
317     </wsp:Policy>
318   </B>
319 </wsp:Policy>
```

320
321 Ideally the x:ContainsPolicy attribute will, at some point, be moved in the WS-Policy
322 namespace.

323 3.1.3 Nesting Policy Processing Rules

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

- 325 1. Assertions MUST specify whether or not they contain nested policy.
- 326 2. Assertions SHOULD specify which other assertions can be present in their nested policy.
- 327 3. Nested assertions need to be specifically designed for nesting inside one or more outer
328 assertions. Assertions SHOULD specify which assertions they can be nested within.
- 329 4. Assertions from one domain SHOULD NOT be nested inside assertions from another domain.
330 For example, assertions from a transaction domain should not be nested inside an assertion from
331 a security domain.
- 332 5. Assertions containing nested policy are normalized recursively such that in the normal form each
333 nested policy contains no choices. Thus each outer assertion that contains nested policy
334 containing choices is duplicated such that there are as many instances of the outer assertion as
335 there are choices in the nested policy, one instance for each nested choice, recursively. See
336 Section 4.1.4 for a worked example of normalization.
- 337 6. Nested policies are intersected in their own processing contexts with the corresponding nested
338 policy in a matching outer assertion. Thus two assertions having nested policy intersect if the
339 outer assertion QName matches and the nested policies intersect. Intersection always occurs
340 using the normalized form. See Section 4.1.5 for a worked example of intersection.
- 341 7. An assertion with an empty nested policy does not intersect with the same assertion without
342 nested policy.

343 3.1.4 Nested Policy Normalization Worked Example

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

```
346 <wsp:Policy>
347   <wsp:ExactlyOne>
348     <wsp:All>
349       <A />
350       <B>
351         <wsp:Policy>
352           <wsp:ExactlyOne>
353             <wsp:All>
354               <C />
355             </wsp:All>
356           <wsp:All>
357             <D />
358           </wsp:All>
359         </wsp:ExactlyOne>
360       </wsp:Policy>
361     </B>
362   </wsp:All>
363 </wsp:ExactlyOne>
364 </wsp:Policy>
```

365
366 The above policy is normalized by, in this case, creating two alternatives, both containing an A assertion
367 and a B assertion. One alternative contains a B assertion with a nested C assertion while the other
368 contains a B assertion with a nested D assertion (normalized form);
369

```
370 <wsp:Policy>  
371   <wsp:ExactlyOne>  
372     <wsp:All>  
373       <A/>  
374       <B>  
375         <wsp:Policy>  
376           <wsp:ExactlyOne>  
377             <wsp:All>  
378               <C/>  
379             </wsp:All>  
380           </wsp:ExactlyOne>  
381         </wsp:Policy>  
382       </B>  
383     </wsp:All>  
384     <wsp:All>  
385       <A/>  
386       <B>  
387         <wsp:Policy>  
388           <wsp:ExactlyOne>  
389             <wsp:All>  
390               <D/>  
391             </wsp:All>  
392           </wsp:ExactlyOne>  
393         </wsp:Policy>  
394       </B>  
395     </wsp:All>  
396   </wsp:ExactlyOne>  
397 </wsp:Policy>
```

398 **3.1.5 Nested Policy Intersection Worked Example**

399 This section shows a worked example of computing the intersection of two policies that contain assertions
400 with nested policy.
401


```

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

```

```

430
431 <wsp:Policy>
432   <wsp:ExactlyOne>
433     <wsp:All>
434       <A />
435       <B>
436         <wsp:Policy>
437           <wsp:ExactlyOne>
438             <wsp:All>
439               <C/>
440             </wsp:All>
441           </wsp:ExactlyOne>
442         </wsp:Policy>
443       </B>
444     </wsp:All>
445   <wsp:All>
446     <A />
447     <B>
448       <wsp:Policy>
449         <wsp:ExactlyOne>
450           <wsp:All>
451             <E/>
452           </wsp:All>
453         </wsp:ExactlyOne>
454       </wsp:Policy>
455     </B>
456   </wsp:All>
457 </wsp:ExactlyOne>
458 </wsp:Policy>

```

459
460 The two policies above, which are already in normal form, are intersected as follows; firstly the QNames
461 of the A and B assertions are intersected then the QNames of the nested assertions inside the B
462 assertions are intersected. In the nested case, only the two B assertions that have nested C assertions
463 match. Thus the intersection of the nested policy is (intersected policy;

464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491

```
<wsp:Policy>
  <wsp:ExactlyOne>
    <wsp:All>
      <A/>
      <A/>
      <B>
        <wsp:Policy>
          <wsp:ExactlyOne>
            <wsp:All>
              <C/>
            </wsp:All>
          </wsp:ExactlyOne>
        </wsp:Policy>
      </B>
      <B>
        <wsp:Policy>
          <wsp:ExactlyOne>
            <wsp:All>
              <C/>
            </wsp:All>
          </wsp:ExactlyOne>
        </wsp:Policy>
      </B>
    </wsp:All>
  </wsp:ExactlyOne>
</wsp:Policy>
```

3.2 Policy Subjects

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

[Message Policy Subject]

This property identifies a Message Policy Subject [WS-PolicyAttachment]. WS-PolicyAttachment defines seven WSDL [WSDL 1.1] policy attachment points with Message Policy Subject:

- wsdl:message
A policy expression containing one or more assertions with Message Policy Subject MUST NOT be attached to a wsdl:message.
- wsdl:portType/wsdl:operation/wsdl:input, ./wsdl:output, or ./wsdl:fault
A policy expression containing one or more assertions with Message Policy Subject MUST NOT be attached to a descendant of wsdl:portType.
- wsdl:binding/wsdl:operation/wsdl:input, ./wsdl:output, or ./wsdl:fault
A policy expression containing one or more of the assertions with Message Policy Subject MUST be attached to a descendant of wsdl:binding.

[Operation Policy Subject]

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

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

518 wsdl:binding/wsdl:operation
519 A policy expression containing one or
520 more token assertions MUST be attached to a wsdl:binding/wsdl:operation.
521

522 **[Endpoint Policy Subject]**
523 A token assertion instance with Endpoint Policy Subject indicates usage of the token for the
524 entire set of messages described for the endpoint:

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

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

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

534 4 Protection Assertions

535 The following assertions are used to identify *what* is being protected and the level of
536 protection provided. These assertions SHOULD apply to [Message Policy Subject]. Note that
537 when assertions defined in this section are present in a policy, the order of those assertions
538 in that policy has no effect on the order of signature and encryption operations (see Section
539 7.3).

540 4.1 Integrity Assertions

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

544 4.1.1 SignedParts Assertion

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

550

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

556 Syntax

```
557 <sp:SignedParts ... >  
558   <sp:Body />?  
559   <sp:Header Name="xs:NCName"? Namespace="xs:anyURI" ... />*  
560   ...  
561 </sp:SignedParts>
```

562

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

564 /sp:SignedParts

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

569 /sp:SignedParts/sp:Body

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

573 /sp:SignedParts/sp:Header

574 Presence of this optional element indicates a specific SOAP header (or set of such
575 headers) needs to be protected. There may be multiple sp:Header elements within a

576 single sp:SignedParts element. If multiple SOAP headers with the same local name
577 but different namespace names are to be integrity protected multiple sp:Header
578 elements are needed, either as part of a single sp:SignedParts assertion or as part of
579 separate sp:SignedParts assertions.

580 /sp:SignedParts/sp:Header/@Name

581 This optional attribute indicates the local name of the SOAP header to be integrity
582 protected. If this attribute is not specified, all SOAP headers whose namespace
583 matches the Namespace attribute are to be protected.

584 /sp:SignedParts/sp:Header/@Namespace

585 This required attribute indicates the namespace of the SOAP header(s) to be integrity
586 protected.

587 4.1.2 SignedElements Assertion

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

593

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

597 Syntax

```
598 <sp:SignedElements XPathVersion="xs:anyURI"? ... >  
599   <sp:XPath>xs:string</sp:XPath>+  
600   ...  
601 </sp:SignedElements>
```

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

603 /sp:SignedElements

604 This assertion specifies the parts of the message that need integrity protection. If no
605 child elements are specified, all message headers targeted at the UltimateReceiver
606 role and the body of the message MUST be integrity protected.

607 /sp:SignedElements/@XPathVersion

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

609 /sp:SignedElements/sp:XPath

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

615 4.2 Confidentiality Assertions

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

619 4.2.1 EncryptedParts Assertion

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

625
626 There MAY be multiple EncryptedParts assertions present. Multiple EncryptedParts
627 assertions present within a policy alternative are equivalent to a single EncryptedParts
628 assertion containing the union of all specified message parts. Note that this assertion does
629 not require that a given part appear in a message, just that if such a part appears, it
630 requires confidentiality protection.

631 Syntax

```
632 <sp:EncryptedParts ... >  
633   <sp:Body/>?  
634   <sp:Header Name="xs:NCName"? Namespace="xs:anyURI" ... />*  
635   ...  
636 </sp:EncryptedParts>
```

637
638 The following describes the attributes and elements listed in the schema outlined above:
639 /sp:EncryptedParts

640 This assertion specifies the parts of the message that need confidentiality protection.
641 The single child element of this assertion specifies the set of message parts using an
642 extensible dialect.

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

645 /sp:EncryptedParts/sp:Body

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

650 /sp:EncryptedParts/sp:Header

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

660 /sp:EncryptedParts/sp:Header/@Name

661 This optional attribute indicates the local name of the SOAP header to be
662 confidentiality protected. If this attribute is not specified, all SOAP headers whose
663 namespace matches the Namespace attribute are to be protected.

664 /sp:EncryptedParts/sp:Header/@Namespace

665 This required attribute indicates the namespace of the SOAP header(s) to be
666 confidentiality protected.

667 **4.2.2 EncryptedElements Assertion**

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

673

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

677 **Syntax**

```
678 <sp:EncryptedElements XPathVersion="xs:anyURI"? ... >  
679   <sp:XPath>xs:string</sp:XPath>+  
680   ...  
681 </sp:EncryptedElements>
```

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

683 /sp:EncryptedElements

684 This assertion specifies the parts of the message that need confidentiality protection.
685 If no child elements are specified, the body of the message MUST be confidentiality
686 protected.

687 /sp:EncryptedElements/@XPathVersion

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

689 /sp:EncryptedElements/sp:XPath

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

694 **4.3 Required Elements Assertion**

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

697

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

701 **4.3.1 RequiredElements Assertion**

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

704

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

708 **Syntax**

```
709 <sp:RequiredElements XPathVersion="xs:anyURI"? ... >  
710   <sp:XPath>xs:string</sp:XPath>+  
711   ...  
712 </sp:RequiredElements>
```

713
714 The following describes the attributes and elements listed in the schema outlined above:

715 /sp:RequiredElements

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

717 /sp:RequiredElements/@XPathVersion

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

719 /sp:RequiredElements/sp:XPath

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

725

5 Token Assertions

726

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

727

which contain them SHOULD recommend a policy attachment point. With the exception of

728

transport token assertions, the token assertions defined in this section are not specific to

729

730

any particular security binding.

731

5.1 Token Inclusion

732

Any token assertion may also carry an optional `sp:IncludeToken` attribute. The schema

733

type of this attribute is `xs:anyURI`. This attribute indicates whether the token should be

734

included, that is written, in the message or whether cryptographic operations utilize an

735

external reference mechanism to refer to the key represented by the token. This attribute is

736

defined as a global attribute in the WS-SecurityPolicy namespace and is intended to be used

737

by any specification that defines token assertions.

738

5.1.1 Token Inclusion Values

739

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

740

http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/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://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/Once	The token MUST be included in only one message sent from initiator to 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://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/AlwaysToRecipient	The token MUST be included in all messages sent from initiator to recipient. The token MUST NOT be include in messages sent from the recipient to the initiator.
http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/Always	The token MUST be included in all messages sent between the initiator and the recipient. This is the default behavior.

741

742

Note: In examples, the namespace URI is replaced with "...". For example, .../IncludeToken/Never is actually

743

<http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/Never>. Other token

744

inclusion URI values MAY be defined but are out-of-scope of this specification.

745

746

The default behavior characteristics defined by this specification if this attribute is not

747

specified on a token assertion are .../IncludeToken/Always.

748 5.2 Token Properties

749 5.2.1 [Derived Keys] Property

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

755 5.3 Token Assertions

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

757 5.3.1 UsernameToken Assertion

758 This element represents a requirement to include a username token. The default version of
759 this token is the wsse:UsernameToken as defined in [WSS: Username Token Profile 1.0].

760 Syntax

```
761 <sp:UsernameToken sp:IncludeToken="xs:anyURI"? ... >  
762   <wsp:Policy>  
763     (  
764       <sp:WssUsernameToken10 ... /> |  
765       <sp:WssUsernameToken11 ... />  
766     ) ?  
767     ...  
768   </wsp:Policy> ?  
769   ...  
770 </sp:UsernameToken>
```

771
772 The following describes the attributes and elements listed in the schema outlined above:

773 /sp:UsernameToken

774 This identifies a UsernameToken assertion.

775 /sp:UsernameToken/@sp:IncludeToken

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

777 /sp:UsernameToken/wsp:Policy

778 This optional element identifies additional requirements for use of the sp:UsernameToken
779 assertion.

780 /sp:UsernameToken/wsp:Policy/sp:WssUsernameToken10

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

783 /sp:UsernameToken/wsp:Policy/sp:WssUsernameToken11

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

786

787 Note: While Username tokens could be used cryptographically, such usage is discouraged in
788 general because of the relatively low entropy typically associated with passwords. This
789 specification does not define a cryptographic binding for the Username token. A new token
790 assertion could be defined to allow for cryptographic binding.

791 5.3.2 IssuedToken Assertion

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

796 Syntax

```
797 <sp:IssuedToken sp:IncludeToken="xs:anyURI"? ... >  
798   <sp:Issuer>wsa:EndpointReferenceType</sp:Issuer?>  
799   <sp:RequestSecurityTokenTemplate TrustVersion="xs:anyURI"? >  
800     ...  
801   </sp:RequestSecurityTokenTemplate>  
802   <wsp:Policy>  
803     <sp:RequireDerivedKeys ... /> ?  
804     <sp:RequireExternalReference ... /> ?  
805     <sp:RequireInternalReference ... /> ?  
806     ...  
807   </wsp:Policy? >  
808   ...  
809 </sp:IssuedToken>
```

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

811 /sp:IssuedToken

812 This identifies an IssuedToken assertion.

813 /sp:IssuedToken/@sp:IncludeToken

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

815 /sp:IssuedToken/sp:Issuer

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

818 /sp:IssuedToken/sp:RequestSecurityTokenTemplate

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

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

822 /sp:IssuedToken/sp:RequestSecurityTokenTemplate/@TrustVersion

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

825 /sp:IssuedToken/wsp:Policy

826 This optional element identifies additional requirements for use of the sp:IssuedToken assertion.

827 /sp:IssuedToken/wsp:Policy/sp:RequireDerivedKeys

828 This optional element sets the [Derived Keys] property for this token to 'true'.

829 /sp:IssuedToken/wsp:Policy/sp:RequireInternalReference

830 This optional element indicates whether an internal reference is required when referencing this
831 token.

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

833 /sp:IssuedToken/wsp:Policy/sp:RequireExternalReference

834 This optional element indicates whether an external reference is required when referencing this
835 token.

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

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

842 5.3.3 X509Token Assertion

843 This element represents a requirement for a binary security token carrying an X509 token.
844 The default version of this token and associated profile is the X509 Version 3 token as
845 specified in [WSS: X509 Certificate Token Profile 1.0].

846 Syntax

```
847 <sp:X509Token sp:IncludeToken="xs:anyURI"? ... >  
848   <wsp:Policy>  
849     <sp:RequireKeyIdentifierReference ... /> ?  
850     <sp:RequireIssuerSerialReference ... /> ?  
851     <sp:RequireEmbeddedTokenReference ... /> ?  
852     <sp:RequireThumbprintReference ... /> ?  
853     (  
854       <sp:WssX509V1Token10 ... /> |  
855       <sp:WssX509V3Token10 ... /> |  
856       <sp:WssX509Pkcs7Token10 ... /> |  
857       <sp:WssX509PkiPathV1Token10 ... /> |  
858       <sp:WssX509V1Token11 ... /> |  
859       <sp:WssX509V3Token11 ... /> |  
860       <sp:WssX509Pkcs7Token11 ... /> |  
861       <sp:WssX509PkiPathV1Token11 ... />  
862     ) ?  
863     ...  
864   </wsp:Policy> ?  
865   ...  
866 </sp:X509Token>
```

867
868 The following describes the attributes and elements listed in the schema outlined above:

869 `/sp:X509Token`

870 This identifies an X509Token assertion.

871 `/sp:X509Token/@sp:IncludeToken`

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

873 `/sp:X509Token/wsp:Policy`

874 This optional element identifies additional requirements for use of the `sp:X509Token` assertion.

875 `/sp:X509Token/wsp:Policy/sp:RequireKeyIdentifierReference`

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

878 `/sp:X509Token/wsp:Policy/sp:RequireIssuerSerialReference`

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

881 `/sp:X509Token/wsp:Policy/sp:RequireEmbeddedTokenReference`

882 This optional element indicates that an embedded token reference is required when referencing
883 this token.

884 `/sp:X509Token/wsp:Policy/sp:RequireThumbprintReference`

885 This optional element indicates that a thumbprint reference is required when referencing this
886 token.

887 /sp:X509Token/wsp:Policy/sp:WssX509V1Token10

888 This optional element indicates that an X509 Version 1 token should be used as defined in [WSS:
889 X509 Token Profile 1.0].

890 /sp:X509Token/wsp:Policy/sp:WssX509V3Token10

891 This optional element indicates that an X509 Version 3 token should be used as defined in [WSS:
892 X509 Token Profile 1.0].

893 /sp:X509Token/wsp:Policy/sp:WssX509Pkcs7Token10

894 This optional element indicates that an X509 PKCS7 token should be used as defined in [WSS:
895 X509 Token Profile 1.0].

896 /sp:X509Token/wsp:Policy/sp:WssX509PkiPathV1Token10

897 This optional element indicates that an X509 PKI Path Version 1 token should be used as defined
898 in [WSS: X509 Token Profile 1.0].

899 /sp:X509Token/wsp:Policy/sp:WssX509V1Token11

900 This optional element indicates that an X509 Version 1 token should be used as defined in [WSS:
901 X509 Token Profile 1.1].

902 /sp:X509Token/wsp:Policy/sp:WssX509V3Token11

903 This optional element indicates that an X509 Version 3 token should be used as defined in [WSS:
904 X509 Token Profile 1.1].

905 /sp:X509Token/wsp:Policy/sp:WssX509Pkcs7Token11

906 This optional element indicates that an X509 PKCS7 token should be used as defined in [WSS:
907 X509 Token Profile 1.1].

908 /sp:X509Token/wsp:Policy/sp:WssX509PkiPathV1Token11

909 This optional element indicates that an X509 PKI Path Version 1 token should be used as defined
910 in [WSS: X509 Token Profile 1.1].

911 **5.3.4 KerberosToken Assertion**

912 This element represents a requirement for a Kerberos token. The default version of this
913 token and associated profile is the Kerberos Version 5 AP-REQ security token as specified in
914 [WSS: Kerberos Token Profile 1.0].

915 **Syntax**

```
916 <sp:KerberosToken sp:IncludeToken="xs:anyURI"? ... >  
917   <wsp:Policy>  
918     <sp:RequireDerivedKeys ... /> ?  
919     <sp:RequireKeyIdentifierReference ... /> ?  
920     (  
921       <sp:WssKerberosV5ApReqToken11 ... /> |  
922       <sp:WssGssKerberosV5ApReqToken11 ... />  
923     ) ?  
924     ...  
925   </wsp:Policy> ?  
926   ...  
927 </sp:KerberosToken>
```

929
930 The following describes the attributes and elements listed in the schema outlined above:

931 /sp:KerberosToken
 932 This identifies a KerberosV5ApReqToken assertion.
 933 /sp:KerberosToken/@sp:IncludeToken
 934 This optional attribute identifies the token inclusion value for this token assertion.
 935 /sp:KerberosToken/wsp:Policy
 936 This optional element identifies additional requirements for use of the sp:KerberosToken
 937 assertion.
 938 /sp:KerberosToken/wsp:Policy/sp:RequireDerivedKeys
 939 This optional element sets the [Derived Keys] property for this token to 'true'.
 940 /sp:KerberosToken/wsp:Policy/sp:RequireKeyIdentifierReference
 941 This optional element indicates that a key identifier reference is required when referencing this
 942 token.
 943 /sp:KerberosToken/wsp:Policy/sp:WssKerberosV5ApReqToken11
 944 This optional element indicates that a Kerberos Version 5 AP-REQ token should be used as
 945 defined in [WSS: Kerberos Token Profile 1.1].
 946 /sp:KerberosToken/wsp:Policy/sp:WssGssKerberosV5ApReqToken11
 947 This optional element indicates that a GSS Kerberos Version 5 AP-REQ token should be used as
 948 defined in [WSS: Kerberos Token Profile 1.1].

949 **5.3.5 SpnegoContextToken Assertion**

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

953 **Syntax**

```

954 <sp:SpnegoContextToken sp:IncludeToken="xs:anyURI"? ... >
955   <sp:Issuer>wsa:EndpointReferenceType</sp:Issuer> ?
956   <wsp:Policy>
957     <sp:RequireDerivedKeys ... /> ?
958     ...
959   </wsp:Policy> ?
960   ...
961 </sp:SpnegoContextToken>
  
```

962
 963 The following describes the attributes and elements listed in the schema outlined above:

964 /sp:SpnegoContextToken
 965 This identifies a SpnegoContextToken assertion.
 966 /sp:SpnegoContextToken/@sp:IncludeToken
 967 This optional attribute identifies the token inclusion value for this token assertion.
 968 /sp:SpnegoContextToken/sp:Issuer
 969 This optional element, of type wsa:EndpointReferenceType, contains a reference to the issuer for the
 970 Spnego Context Token.
 971 /sp:SpnegoContextToken/wsp:Policy
 972 This optional element identifies additional requirements for use of the sp:SpnegoContextToken
 973 assertion.

974 /sp:SpnegoContextToken/wsp:Policy/sp:RequireDerivedKeys
975 This optional element sets the [Derived Keys] property for this token to 'true'.

976 **5.3.6 SecurityContextToken Assertion**

977 This element represents a requirement for a SecurityContextToken token. The default
978 version of this token is the Security Context Token as specified in [WS-SecureConversation
979 1.0].

980 **Syntax**

```
981 <sp:SecurityContextToken sp:IncludeToken="xs:anyURI"? ... >  
982 <wsp:Policy>  
983 <sp:RequireDerivedKeys ... /> ?  
984 <sp:RequireExternalUriReference ... /> ?  
985 <sp:SC10SecurityContextToken ... /> ?  
986 ...  
987 </wsp:Policy> ?  
988 ...  
989 </sp:SecurityContextToken>
```

990
991 The following describes the attributes and elements listed in the schema outlined above:

992 /sp:SecurityContextToken

993 This identifies a SecurityContextToken assertion.

994 /sp:SecurityContextToken/@sp:IncludeToken

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

996 /sp:SecurityContextToken/wsp:Policy

997 This optional element identifies additional requirements for use of the sp:SecurityContextToken
998 assertion.

999 /sp:SecurityContextToken/wsp:Policy/sp:RequireDerivedKeys

1000 This optional element sets the [Derived Keys] property for this token to 'true'.

1001 /sp:SecurityContextToken/wsp:Policy/sp:RequireExternalUriReference

1002 This optional element indicates that an external URI reference is required when referencing this
1003 token.

1004 /sp:SecurityContextToken/wsp:Policy/sp:SC10SecurityContextToken

1005 This optional element indicates that a Security Context Token should be used as defined in [WS-
1006 SecureConversation 1.0].

1007

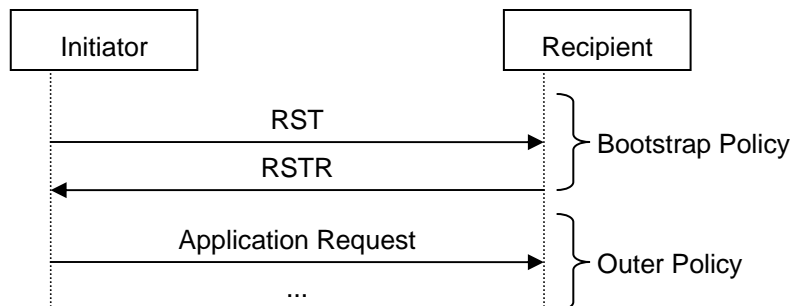
1008 Note: This assertion does not describe how to obtain a Security Context Token but rather
1009 assumes that both parties have the token already or have agreed separately on a
1010 mechanism for obtaining the token. If a definition of the mechanism for obtaining the
1011 Security Context Token is desired in policy, then either the sp:SecureConversationToken or
1012 the sp:IssuedToken assertion should be used instead.

1013 **5.3.7 SecureConversationToken Assertion**

1014 This element represents a requirement for a Security Context Token retrieved from the
1015 indicated issuer address. The default version of this token and associated protocol is the
1016 Security Context Token as defined in [WS-SecureConversation 1.0]. If the sp:Issuer address

1017 is absent, the protocol MUST be executed at the same address as the service endpoint
1018 address.

1019
1020 Note: This assertion describes the token accepted by the target service. Because this token
1021 is issued by the target service and may not have a separate port (with separate policy), this
1022 assertion SHOULD contain a bootstrap policy indicating the security binding and policy that
1023 is used when requesting this token from the target service. That is, the bootstrap policy is
1024 used to obtain the token and then the current (outer) policy is used when making requests
1025 with the token. This is illustrated in the diagram below.



1026

1027 Syntax

```
1028 <sp:SecureConversationToken sp:IncludeToken="xs:anyURI"? ... >
1029   <sp:Issuer>wsa:EndpointReferenceType</sp:Issuer>?
1030   <wsp:Policy>
1031     <sp:RequireDerivedKeys ... /> ?
1032     <sp:RequireExternalUriReference ... /> ?
1033     <sp:SC10SecurityContextToken ... /> ?
1034     <sp:BootstrapPolicy ... > ?
1035     <wsp:Policy> ... </wsp:Policy>
1036   </sp:BootstrapPolicy>
1037 </wsp:Policy> ?
1038   ...
1039 </sp:SecureConversationToken>
```

1040

1041 The following describes the attributes and elements listed in the schema outlined above:

1042 /sp:SecureConversationToken

1043 This identifies a SecureConversationToken assertion.

1044 /sp:SecureConversationToken/@sp:IncludeToken

1045 This optional attribute identifies the token inclusion value for this token assertion.

1046 /sp:SecureConversationToken/sp:Issuer

1047 This optional element, of type wsa:EndpointReferenceType, contains a reference to the issuer for the
1048 Security Context Token.

1049 /sp:SecureConversationToken/wsp:Policy

1050 This optional element identifies additional requirements for use of the
1051 sp:SecureConversationToken assertion.

1052 /sp:SecureConversationToken/wsp:Policy/sp:RequireDerivedKeys

1053 This optional element sets the [Derived Keys] property for this token to 'true'.

1054 /sp:SecureConversationToken/wsp:Policy/sp:RequireExternalUriReference

1055 This optional element indicates that an external URI reference is required when referencing this
1056 token.

1057 /sp:SecureConversationToken/wsp:Policy/sp:SC10SecurityContextToken

1058 This optional element indicates that a Security Context Token should be used as obtained using
1059 the protocol defined in [WS-SecureConversation 1.0].

1060 /sp:SecureConversationToken/wsp:Policy/sp:BootstrapPolicy

1061 This optional element contains the policy indicating the requirements for obtaining the Security
1062 Context Token.

1063 /sp:SecureConversationToken/wsp:Policy/sp:BootstrapPolicy/wsp:Policy

1064 This element contains the security binding requirements for obtaining the Security Context Token.

1065 Example

```
1066 <wsp:Policy>  
1067   <sp:SymmetricBinding>  
1068     <wsp:Policy>  
1069       <sp:ProtectionToken>  
1070         <wsp:Policy>  
1071           <sp:SecureConversationToken>  
1072             <sp:Issuer>  
1073               <wsa:Address>http://example.org/sts</wsa:Address>  
1074             </sp:Issuer>  
1075           <wsp:Policy>  
1076             <sp:SC10SecurityContextToken />  
1077             <sp:BootstrapPolicy>  
1078               <wsp:Policy>  
1079                 <sp:AsymmetricBinding>  
1080                   <wsp:Policy>  
1081                     <sp:InitiatorToken>  
1082                       ...  
1083                     </sp:InitiatorToken>  
1084                     <sp:RecipientToken>  
1085                       ...  
1086                     </sp:RecipientToken>  
1087                   </wsp:Policy>  
1088                 </sp:AsymmetricBinding>  
1089               <sp:SignedParts>  
1090                 ...  
1091               </sp:SignedParts>  
1092             </wsp:Policy>  
1093           </sp:BootstrapPolicy>  
1094         </wsp:Policy>  
1095       </sp:SecureConversationToken>  
1096     </wsp:Policy>  
1097   </sp:ProtectionToken>  
1098   ...  
1099 </wsp:Policy>  
1100 </sp:SymmetricBinding>  
1101 <sp:SignedParts>  
1102   ...  
1103 </sp:SignedParts>  
1104   ...  
1105 </wsp:Policy>
```

1107 5.3.8 SamlToken Assertion

1108 This element represents a requirement for a SAML token. The default version of this token
1109 and associated profile is SAML Version 1.0 token as described in the [WSS: SAML Token
1110 Profile].

1111 **Syntax**

```
1112 <sp:SamlToken sp:IncludeToken="xs:anyURI"? ... >
1113   <wsp:Policy>
1114     <sp:RequireDerivedKeys ... /> ?
1115     <sp:RequireKeyIdentifierReference ... /> ?
1116     (
1117       <sp:WssSamlV10Token10 ... /> |
1118       <sp:WssSamlV11Token10 ... /> |
1119       <sp:WssSamlV10Token11 ... /> |
1120       <sp:WssSamlV11Token11 ... /> |
1121       <sp:WssSamlV20Token11 ... />
1122     ) ?
1123     ...
1124   </wsp:Policy> ?
1125   ...
1126 </sp:SamlToken>
```

1127

1128 The following describes the attributes and elements listed in the schema outlined above:

1129 /sp:SamlToken

1130 This identifies a SamlToken assertion.

1131 /sp:SamlToken/@sp:IncludeToken

1132 This optional attribute identifies the token inclusion value for this token assertion.

1133 /sp:SamlToken/wsp:Policy

1134 This optional element identifies additional requirements for use of the sp:SamlToken assertion.

1135 /sp:SamlToken/wsp:Policy/sp:RequireDerivedKeys

1136 This optional element sets the [Derived Keys] property for this token to 'true'.

1137 /sp:SamlToken/wsp:Policy/sp:RequireKeyIdentifierReference

1138 This optional element indicates that a key identifier reference is required when referencing this
1139 token.

1140 /sp:SamlToken/wsp:Policy/sp:WssSamlV10Token10

1141 This optional element identifies that a SAML Version 1.0 token should be used as defined in
1142 [WSS: SAML Token Profile 1.0].

1143 /sp:SamlToken/wsp:Policy/sp:WssSamlV11Token10

1144 This optional element identifies that a SAML Version 1.1 token should be used as defined in
1145 [WSS: SAML Token Profile 1.0].

1146 /sp:SamlToken/wsp:Policy/sp:WssSamlV10Token11

1147 This optional element identifies that a SAML Version 1.0 token should be used as defined in
1148 [WSS: SAML Token Profile 1.1].

1149 /sp:SamlToken/wsp:Policy/sp:WssSamlV11Token11

1150 This optional element identifies that a SAML Version 1.1 token should be used as defined in
1151 [WSS: SAML Token Profile 1.1].

1152 /sp:SamlToken/wsp:Policy/sp:WssSamlV20Token11

1153 This optional element identifies that a SAML Version 2.0 token should be used as defined in
1154 [WSS: SAML Token Profile 1.1].

1155

1156 Note: This assertion does not describe how to obtain a SAML Token but rather assumes that
1157 both parties have the token already or have agreed separately on a mechanism for
1158 obtaining the token. If a definition of the mechanism for obtaining the SAML Token is
1159 desired in policy, the sp:IssuedToken assertion should be used instead.

1160 **5.3.9 RelToken Assertion**

1161 This element represents a requirement for a REL token. The default version of this token
1162 and associate profile is the REL Version 1.0 token as described in the [WSS: REL Token
1163 Profile].

1164 **Syntax**

```
1165 <sp:RelToken sp:IncludeToken="xs:anyURI"? ... >  
1166   <wsp:Policy>  
1167     <sp:RequireDerivedKeys ... /> ?  
1168     <sp:RequireKeyIdentifierReference ... /> ?  
1169     (  
1170       <sp:WssRelV10Token10 ... /> |  
1171       <sp:WssRelV20Token10 ... /> |  
1172       <sp:WssRelV10Token11 ... /> |  
1173       <sp:WssRelV20Token11 ... />  
1174     ) ?  
1175     ...  
1176   </wsp:Policy> ?  
1177   ...  
1178 </sp:RelToken>
```

1179
1180 The following describes the attributes and elements listed in the schema outlined above:
1181 /sp:RelToken

1182 This identifies a RelToken assertion.

1183 /sp:RelToken/@sp:IncludeToken

1184 This optional attribute identifies the token inclusion value for this token assertion.

1185 /sp:RelToken/wsp:Policy

1186 This optional element identifies additional requirements for use of the sp:RelToken assertion.

1187 /sp:RelToken/wsp:Policy/sp:RequireDerivedKeys

1188 This optional element sets the [Derived Keys] property for this token to 'true'.

1189 /sp:RelToken/wsp:Policy/sp:RequireKeyIdentifierReference

1190 This optional element indicates that a key identifier reference is required when referencing this
1191 token.

1192 /sp:RelToken/wsp:Policy/sp:WssRelV10Token10

1193 This optional element identifies that a REL Version 1.0 token should be used as defined in [WSS:
1194 REL Token Profile 1.0].

1195 /sp:RelToken/wsp:Policy/sp:WssRelV20Token10

1196 This optional element identifies that a REL Version 2.0 token should be used as defined in [WSS:
1197 REL Token Profile 1.0].

1198 /sp:RelToken/wsp:Policy/sp:WssRelV10Token11

1199 This optional element identifies that a REL Version 1.0 token should be used as defined in [WSS:
1200 REL Token Profile 1.1].

1201 /sp:RelToken/wsp:Policy/sp:WssRelV20Token11

1202 This optional element identifies that a REL Version 2.0 token should be used as defined in [WSS:
1203 REL Token Profile 1.1].

1204

1205 Note: This assertion does not describe how to obtain a REL Token but rather assumes that
1206 both parties have the token already or have agreed separately on a mechanism for
1207 obtaining the token. If a definition of the mechanism for obtaining the REL Token is desired
1208 in policy, the sp:IssuedToken assertion should be used instead.

1209 5.3.10 HttpsToken Assertion

1210 This element represents a requirement for a transport binding to support the use of HTTPS.

1211 Syntax

```
1212 <sp:HttpsToken RequireClientCertificate="xs:boolean" ... >  
1213   <wsp:Policy>  
1214     ...  
1215   </wsp:Policy> ?  
1216   ...  
1217 </sp:HttpsToken>
```

1218 The following describes the attributes and elements listed in the schema outlined above:

1219 /sp:HttpsToken

1220 This identifies an Https assertion stating that use of the HTTPS protocol specification is
1221 supported.

1222 /sp:HttpsToken/@RequireClientCertificate

1223 The client MUST provide a certificate when negotiating the HTTPS session.

1224 /sp:HttpsToken/wsp:Policy

1225 This optional element identifies additional requirements for use of the sp:HttpsToken assertion.

1226 6 Security Binding Properties

1227 This section defines the various properties or conditions of a security binding, their
1228 semantics, values and defaults where appropriate. Properties are used by a binding in a
1229 manner similar to how variables are used in code. Assertions populate, (or set) the value of
1230 the property (or variable). When an assertion that populates a value of a property appears
1231 in a policy, that property is set to the value indicated by the assertion. The security binding
1232 then uses the value of the property to control its behavior. The properties listed here are
1233 common to the various security bindings described in Section 8. Assertions that define
1234 values for these properties are defined in Section 8. The following properties are used by
1235 the security binding assertions.

1236 6.1 [Algorithm Suite] Property

1237 This property specifies the algorithm suite required for performing cryptographic operations
1238 with symmetric or asymmetric key based security tokens. An algorithm suite specifies actual
1239 algorithms and allowed key lengths. A policy alternative will define what algorithms are
1240 used and how they are used. This property defines the set of available algorithms. The
1241 value of this property is typically referenced by a security binding and is used to specify the
1242 algorithms used for all cryptographic operations performed under the security binding.

1243 Note: In some cases, this property MAY be referenced under a context other than a security
1244 binding and used to control the algorithms used under that context. For example,
1245 supporting token assertions define such a context.

1246 An algorithm suite defines values for each of the following operations and properties:

- 1247 • [Sym Sig] Symmetric Key Signature
- 1248 • [Asym Sig] Signature with an asymmetric key
- 1249 • [Dig] Digest
- 1250 • [Enc] Encryption
- 1251 • [Sym KW] Symmetric Key Wrap
- 1252 • [Asym KW] Asymmetric Key Wrap
- 1253 • [Comp Key] Computed key
- 1254 • [Enc KD] Encryption key derivation
- 1255 • [Sig KD] Signature key derivation
- 1256 • [Min SKL] Minimum symmetric key length
- 1257 • [Max SKL] Maximum symmetric key length
- 1258 • [Min AKL] Minimum asymmetric key length
- 1259 • [Max AKL] Maximum asymmetric key length

1260

1261 The following table provides abbreviations for the algorithm URI used in the table below:

Abbreviation	Algorithm URI
HmacSha1	http://www.w3.org/2000/09/xmlsig#hmac-sha1
RsaSha1	http://www.w3.org/2000/09/xmlsig#rsa-sha1
Sha1	http://www.w3.org/2000/09/xmlsig#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://schemas.xmlsoap.org/ws/2005/02/sc/dk/p_sha1
PSha1L128	http://schemas.xmlsoap.org/ws/2005/02/sc/dk/p_sha1
PSha1L192	http://schemas.xmlsoap.org/ws/2005/02/sc/dk/p_sha1
PSha1L256	http://schemas.xmlsoap.org/ws/2005/02/sc/dk/p_sha1
XPath	http://www.w3.org/TR/1999/REC-xpath-19991116
XPath20	http://www.w3.org/2002/06/xmldsig-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

1262

1263 The tables below show all the base algorithm suites defined by this specification. This table
 1264 defines values for properties which are common for all suites:

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

1265 This table defines additional properties whose values can be specified along with the default
 1266 value for that property.

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

1267 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

1268 6.2 [Timestamp] Property

1269 This boolean property specifies whether a `wsu:Timestamp` element is present in the
1270 `wsse:Security` header. If the value is 'true', the timestamp element MUST be present and
1271 MUST be integrity protected either by transport or message level security. If the value is
1272 'false', the timestamp element MUST NOT be present. The default value for this property is
1273 'false'.

1274 6.3 [Protection Order] Property

1275 This property indicates the order in which integrity and confidentiality are applied to the
1276 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.

1277 The default value for this property is 'SignBeforeEncrypting'.

1278 6.4 [Signature Protection] Property

1279 This boolean property specifies whether the signature must be encrypted. If the value is
1280 'true', the primary signature MUST be encrypted and any signature confirmation elements
1281 MUST also be encrypted. If the value is 'false', the primary signature MUST NOT be
1282 encrypted and any signature confirmation elements MUST NOT be encrypted. The default
1283 value for this property is 'false'.

1284 **6.5 [Token Protection] Property**

1285 This boolean property specifies whether signatures must cover the token used to generate
1286 that signature. If the value is 'true', then each token used to generate a signature MUST be
1287 covered by that signature. If the value is 'false', then the token MUST NOT be covered by
1288 the signature. Note that in cases where derived keys are used, the 'main' token and NOT
1289 the derived key token is covered by the signature. It is recommended that assertions that
1290 define values for this property apply to [Endpoint Policy Subject]. The default value for this
1291 property is 'false'.

1292 **6.6 [Entire Header and Body Signatures] Property**

1293 This boolean property specifies whether signature digests over the SOAP body and SOAP
1294 headers must only cover the entire body and entire header elements. If the value is 'true',
1295 then each digest over the SOAP body MUST be over the entire SOAP body element and not
1296 a descendant of that element. In addition each digest over a SOAP header MUST be over an
1297 actual header element and not a descendant of a header element. This restriction does not
1298 specifically apply to the wsse:Security header. However signature digests over child
1299 elements of the wsse:Security header MUST be over the entire child element and not a
1300 descendent of that element. If the value is 'false', then signature digests MAY be over a
1301 descendant of the SOAP Body or a descendant of a header element. Setting the value of this
1302 property to 'true' mitigates against some possible re-writing attacks. It is recommended
1303 that assertions that define values for this property apply to [Endpoint Policy Subject]. The
1304 default value for this property is 'false'.

1305 **6.7 [Security Header Layout] Property**

1306 This property indicates which layout rules to apply when adding items to the security
1307 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
LaxTimestampLast	As Lax except that the last item in the security header MUST be a wsse:Timestamp

1308 The default value of this property is 'Lax'.

1309 **6.7.1 Strict Layout Rules**

- 1310 1. Tokens that are included in the message MUST be declared before use. For example,
1311 a. A local signing token MUST occur before the signature that uses it.

- 1312 b. A local token serving as the source token for a derived key token MUST occur before that
1313 derived key token.
- 1314 c. A local encryption token MUST occur before the reference list that points to
1315 `xenc:EncryptedData` elements that use it.
- 1316 d. If the same token is used for both signing and encryption, then it should appear before
1317 the earlier element in the security header.
- 1318 2. Signed elements inside the security header MUST occur before the signature that signs them.
1319 For example,
- 1320 a. A timestamp MUST occur before the signature that signs it.
- 1321 b. A Username token (usually in encrypted form) MUST occur before the signature that
1322 signs it.
- 1323 c. A primary signature MUST occur before the supporting token signature that signs the
1324 primary signature's signature value element.
- 1325 d. A `wssell:SignatureConfirmation` element MUST occur before the signature that
1326 signs it.
- 1327 3. When an element in a security header is encrypted, the resulting `xenc:EncryptedData`
1328 element has the same order requirements as the source plain text element. For example, an
1329 encrypted primary signature MUST occur before any supporting token signature per 2c above
1330 and an encrypted token has the same ordering requirements as the unencrypted token.
- 1331 4. If there are any encrypted elements in the message then a top level `xenc:ReferenceList`
1332 element MUST be present in the security header. The `xenc:ReferenceList` MUST occur before
1333 any `xenc:EncryptedData` elements in the security header that are referenced from the
1334 reference list. However, the `xenc:ReferenceList` is not required to appear before independently
1335 encrypted tokens such as the `xenc:EncryptedKey` token as defined in WSS.
- 1336 5. An `xenc:EncryptedKey` element without an internal reference list [[WSS: SOAP Message](#)
1337 [Security 1.1](#)] MUST obey rule (1). An `xenc:EncryptedKey` element with an internal reference
1338 list MUST additionally obey rule (4).
- 1339 Examples of these layout rules for each security binding are described in Appendix C.

7 Security Binding Assertions

1340

1341 The appropriate representation of the different facets of security mechanisms requires
1342 distilling the common primitives (to enable reuse) and then combining the primitive
1343 elements into patterns.

7.1 AlgorithmSuite Assertion

1345 This assertion indicates a requirement for an algorithm suite as defined under the
1346 [Algorithm Suite] property described in Section 7.1. The scope of this assertion is defined by
1347 its containing assertion.

Syntax

```
1349 <sp:AlgorithmSuite ... >  
1350   <wsp:Policy>  
1351     (<sp:Basic256 ... /> |  
1352     <sp:Basic192 ... /> |  
1353     <sp:Basic128 ... /> |  
1354     <sp:TripleDes ... /> |  
1355     <sp:Basic256Rsa15 ... /> |  
1356     <sp:Basic192Rsa15 ... /> |  
1357     <sp:Basic128Rsa15 ... /> |  
1358     <sp:TripleDesRsa15 ... /> |  
1359     <sp:Basic256Sha256 ... /> |  
1360     <sp:Basic192Sha256 ... /> |  
1361     <sp:Basic128Sha256 ... /> |  
1362     <sp:TripleDesSha256 ... /> |  
1363     <sp:Basic256Sha256Rsa15 ... /> |  
1364     <sp:Basic192Sha256Rsa15 ... /> |  
1365     <sp:Basic128Sha256Rsa15 ... /> |  
1366     <sp:TripleDesSha256Rsa15 ... /> |  
1367     ...)  
1368     <sp:InclusiveC14N ... /> ?  
1369     <sp:SOAPNormalization10 ... /> ?  
1370     <sp:STRTransform10 ... /> ?  
1371     <sp:XPath10 ... /> ?  
1372     <sp:XPathFilter20 ... /> ?  
1373     ...  
1374   </wsp:Policy>  
1375   ...  
1376 </sp:AlgorithmSuite>
```

1377

1378 The following describes the attributes and elements listed in the schema outlined above:

1379 /sp:AlgorithmSuite

1380 This identifies an AlgorithmSuite assertion.

1381 /sp:AlgorithmSuite/wsp:Policy

1382 This element contains one or more policy assertions that indicate the specific algorithm suite to use.

1383 /sp:AlgorithmSuite/wsp:Policy/sp:Basic256

1384 This assertion indicates that the [Algorithm Suite] property is set to 'Basic256'.

1385 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192

1386 This assertion indicates that the [Algorithm Suite] property is set to 'Basic192'.

1387 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128

1388 This assertion indicates that the [Algorithm Suite] property is set to 'Basic128'.
1389 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDes
1390 This assertion indicates that the [Algorithm Suite] property is set to 'TripleDes'.
1391 /sp:AlgorithmSuite/wsp:Policy/sp:Basic256Rsa15
1392 This assertion indicates that the [Algorithm Suite] property is set to 'Basic256Rsa15'.
1393 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192Rsa15
1394 This assertion indicates that the [Algorithm Suite] property is set to 'Basic192Rsa15'.
1395 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128Rsa15
1396 This assertion indicates that the [Algorithm Suite] property is set to 'Basic128Rsa15'.
1397 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDesRsa15
1398 This assertion indicates that the [Algorithm Suite] property is set to 'TripleDesRsa15'.
1399 /sp:AlgorithmSuite/wsp:Policy/sp:Basic256Sha256
1400 This assertion indicates that the [Algorithm Suite] property is set to 'Basic256Sha256'.
1401 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192Sha256
1402 This assertion indicates that the [Algorithm Suite] property is set to 'Basic192Sha256'.
1403 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128Sha256
1404 This assertion indicates that the [Algorithm Suite] property is set to 'Basic128Sha256'.
1405 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDesSha256
1406 This assertion indicates that the [Algorithm Suite] property is set to 'TripleDesSha256'.
1407 /sp:AlgorithmSuite/wsp:Policy/sp:Basic256Sha256Rsa15
1408 This assertion indicates that the [Algorithm Suite] property is set to 'Basic256Sha256Rsa15'.
1409 /sp:AlgorithmSuite/wsp:Policy/sp:Basic192Sha256Rsa15
1410 This assertion indicates that the [Algorithm Suite] property is set to 'Basic192Sha256Rsa15'.
1411 /sp:AlgorithmSuite/wsp:Policy/sp:Basic128Sha256Rsa15
1412 This assertion indicates that the [Algorithm Suite] property is set to 'Basic128Sha256Rsa15'.
1413 /sp:AlgorithmSuite/wsp:Policy/sp:TripleDesSha256Rsa15
1414 This assertion indicates that the [Algorithm Suite] property is set to 'TripleDesSha256Rsa15'.
1415 /sp:AlgorithmSuite/wsp:Policy/sp:InclusiveC14N
1416 This assertion indicates that the [C14N] property of an algorithm suite is set to 'C14N'.
1417 /sp:AlgorithmSuite/wsp:Policy/sp:SoapNormalization10
1418 This assertion indicates that the [SOAP Norm] property is set to 'SNT'.
1419 /sp:AlgorithmSuite/wsp:Policy/sp:STRTransform10
1420 This assertion indicates that the [STR Transform] property is set to 'STRT10'.
1421 /sp:AlgorithmSuite/wsp:Policy/sp:XPath10
1422 This assertion indicates that the [XPath] property is set to 'XPath'.
1423 /sp:AlgorithmSuite/wsp:Policy/sp:XPathFilter20
1424 This assertion indicates that the [XPath] property is set to 'XPath20'.
1425

1426 7.2 Layout Assertion

1427 This assertion indicates a requirement for a particular security header layout as defined
1428 under the [Security Header Layout] property described in Section 7.7. The scope of this
1429 assertion is defined by its containing assertion.

1430 Syntax

```
1431 <sp:Layout ... >  
1432   <wsp:Policy>  
1433     <sp:Strict ... /> |  
1434     <sp:Lax ... /> |  
1435     <sp:LaxTsFirst ... /> |  
1436     <sp:LaxTsLast ... /> |  
1437     ...  
1438   </wsp:Policy>  
1439   ...  
1440 </sp:Layout>
```

1441
1442 The following describes the attributes and elements listed in the schema outlined above:

1443 /sp:Layout

1444 This identifies a Layout assertion.

1445 /sp:Layout/wsp:Policy

1446 This element contains one or more policy assertions that indicate the specific security header layout
1447 to use.

1448 /sp:Layout/wsp:Policy/sp:Strict

1449 This assertion indicates that the [Security Header Layout] property is set to 'Strict'.

1450 /sp:Layout/wsp:Policy/sp:Lax

1451 This assertion indicates that the [Security Header Layout] property is set to 'Lax'.

1452 /sp:Layout/wsp:Policy/sp:LaxTsFirst

1453 This assertion indicates that the [Security Header Layout] property is set to 'LaxTimestampFirst'.

1454 /sp:Layout/wsp:Policy/sp:LaxTsLast

1455 This assertion indicates that the [Security Header Layout] property is set to 'LaxTimestampLast'.

1456 7.3 TransportBinding Assertion

1457 The TransportBinding assertion is used in scenarios in which message protection and
1458 security correlation is provided by means other than [WSS: SOAP Message Security](#), for
1459 example by a secure transport like HTTPS. Specifically, this assertion indicates that the
1460 message is protected using the means provided by the transport. This binding has one
1461 binding specific token property; [Transport Token]. This assertion MUST apply to [Endpoint
1462 Policy Subject].

1463 Syntax

```

1464 <sp:TransportBinding ... >
1465   <wsp:Policy>
1466     <sp:TransportToken ... >
1467       <wsp:Policy> ... </wsp:Policy>
1468       ...
1469     </sp:TransportToken>
1470     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite>
1471     <sp:Layout ... > ... </sp:Layout> ?
1472     <sp:IncludeTimestamp ... /> ?
1473     ...
1474   </wsp:Policy>
1475   ...
1476 </sp:TransportBinding>

```

1477
1478 The following describes the attributes and elements listed in the schema outlined above:

1479 /sp:TransportBinding

1480 This identifies a TransportBinding assertion.

1481 /sp:TransportBinding/wsp:Policy

1482 This indicates a nested `wsp:Policy` element that defines the behavior of the TransportBinding
1483 assertion.

1484 /sp:TransportBinding/wsp:Policy/sp:TransportToken

1485 This assertion indicates a requirement for a Transport Token. The specified token populates the
1486 [Transport Token] property and indicates how the transport is secured.

1487 /sp:TransportBinding/wsp:Policy/sp:TransportToken/wsp:Policy

1488 This indicates a nested policy that identifies the type of Transport Token to use.

1489 /sp:TransportBinding/wsp:Policy/sp:AlgorithmSuite

1490 This assertion indicates a value that populates the [Algorithm Suite] property. See Section 8.1 for
1491 more details.

1492 /sp:TransportBinding/wsp:Policy/sp:Layout

1493 This assertion indicates a value that populates the [Security Header Layout] property. See
1494 Section 8.2 for more details.

1495 /sp:TransportBinding/wsp:Policy/sp:IncludeTimestamp

1496 This assertion indicates that the [Timestamp] property is set to 'true'.

1497 **7.4 SymmetricBinding Assertion**

1498 The SymmetricBinding assertion is used in scenarios in which message protection is
1499 provided by means defined in [WSS: SOAP Message Security](#). This binding has two binding
1500 specific token properties; [Encryption Token] and [Signature Token]. If the message pattern
1501 requires multiple messages, this binding defines that the [Encryption Token] used from
1502 initiator to recipient is also used from recipient to initiator. Similarly, the [Signature Token]
1503 used from initiator to recipient is also use from recipient to initiator. If a `sp:ProtectionToken`
1504 assertion is specified, the specified token populates both token properties and is used as the
1505 basis for both encryption and signature in both directions. This assertion MUST apply to
1506 [Endpoint Policy Subject].

1507 **Syntax**

```

1508 <sp:SymmetricBinding ... >
1509   <wsp:Policy>
1510     (
1511       <sp:EncryptionToken ... >
1512         <wsp:Policy> ... </wsp:Policy>
1513       </sp:EncryptionToken>
1514       <sp:SignatureToken ... >
1515         <wsp:Policy> ... </wsp:Policy>
1516       </sp:SignatureToken>
1517     ) | (
1518       <sp:ProtectionToken ... >
1519         <wsp:Policy> ... </wsp:Policy>
1520       </sp:ProtectionToken>
1521     )
1522   <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite>
1523   <sp:Layout ... > ... </sp:Layout> ?
1524   <sp:IncludeTimestamp ... /> ?
1525   <sp:EncryptBeforeSigning ... /> ?
1526   <sp:EncryptSignature ... /> ?
1527   <sp:ProtectTokens ... /> ?
1528   <sp:OnlySignEntireHeadersAndBody ... /> ?
1529   ...
1530 </wsp:Policy>
1531 ...
1532 </sp:SymmetricBinding>

```

1533
1534 The following describes the attributes and elements listed in the schema outlined above:

1535 /sp:SymmetricBinding

1536 This identifies a SymmetricBinding assertion.

1537 /sp:SymmetricBinding/wsp:Policy

1538 This indicates a nested wsp:Policy element that defines the behavior of the
1539 SymmetricBinding assertion.

1540 /sp:SymmetricBinding/wsp:Policy/sp:EncryptionToken

1541 This assertion indicates a requirement for an Encryption Token. The specified token populates
1542 the [Encryption Token] property and is used for encryption. It is an error for both an
1543 sp:EncryptionToken and an sp:ProtectionToken assertion to be specified.

1544 /sp:SymmetricBinding/wsp:Policy/sp:EncryptionToken/wsp:Policy

1545 The policy contained here MUST identify one or more tokens to use for encryption.

1546 /sp:SymmetricBinding/wsp:Policy/sp:SignatureToken

1547 This assertion indicates a requirement for a Signature Token. The specified token populates the
1548 [Signature Token] property and is used for the message signature. It is an error for both an
1549 sp:SignatureToken and an sp:ProtectionToken assertion to be specified.

1550 /sp:SymmetricBinding/wsp:Policy/sp:SignatureToken/wsp:Policy

1551 The policy contained here MUST identify one or more tokens to use for signatures.

1552 /sp:SymmetricBinding/wsp:Policy/sp:ProtectionToken

1553 This assertion indicates a requirement for a Protection Token. The specified token populates the
1554 [Encryption Token] and [Signature Token properties] and is used for the message signature and
1555 for encryption. It is an error for both an sp:ProtectionToken assertion and either an
1556 sp:EncryptionToken assertion or an sp:SignatureToken assertion to be specified.

1557 /sp:SymmetricBinding/wsp:Policy/sp:ProtectionToken/wsp:Policy

1558 The policy contained here MUST identify exactly one token to use for protection.

1559 /sp:SymmetricBinding/wsp:Policy/sp:AlgorithmSuite
 1560 This assertion indicates a value that populates the [Algorithm Suite] property. See Section 8.1 for
 1561 more details.

1562 /sp:SymmetricBinding/wsp:Policy/sp:Layout
 1563 This assertion indicates a value that populates the [Security Header Layout] property. See
 1564 Section 8.1 for more details.

1565 /sp:SymmetricBinding/wsp:Policy/sp:IncludeTimestamp
 1566 This assertion indicates that the [Timestamp] property is set to 'true'.

1567 /sp:SymmetricBinding/wsp:Policy/sp:EncryptBeforeSigning
 1568 This assertion indicates that the [Protection Order] property is set to 'EncryptBeforeSigning'.

1569 /sp:SymmetricBinding/wsp:Policy/sp:EncryptSignature
 1570 This assertion indicates that the [Signature Protection] property is set to 'true'.

1571 /sp:SymmetricBinding/wsp:Policy/sp:ProtectTokens
 1572 This assertion indicates that the [Token Protection] property is set to 'true'.

1573 /sp:SymmetricBinding/wsp:Policy/sp:OnlySignEntireHeadersAndBody
 1574 This assertion indicates that the [Entire Header And Body Signatures] property is set to 'true'.

1575 7.5 AsymmetricBinding Assertion

1576 The AsymmetricBinding assertion is used in scenarios in which message protection is
 1577 provided by means defined in [WSS: SOAP Message Security](#). This binding has two binding
 1578 specific token properties; [Initiator Token] and [Recipient Token]. If the message pattern
 1579 requires multiple messages, this binding defines that the [Initiator Token] is used for the
 1580 message signature from initiator to the recipient, and for encryption from recipient to
 1581 initiator. The [Recipient Token] is used for encryption from initiator to recipient, and for the
 1582 message signature from recipient to initiator. This assertion MUST apply to [Endpoint Policy
 1583 Subject].

1584 Syntax

```

1585 <sp:AsymmetricBinding ... >
1586   <wsp:Policy>
1587     <sp:InitiatorToken>
1588       <wsp:Policy> ... </wsp:Policy>
1589     </sp:InitiatorToken>
1590     <sp:RecipientToken>
1591       <wsp:Policy> ... </wsp:Policy>
1592     </sp:RecipientToken>
1593     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite>
1594     <sp:Layout ... > ... </sp:Layout> ?
1595     <sp:IncludeTimestamp ... /> ?
1596     <sp:EncryptBeforeSigning ... /> ?
1597     <sp:EncryptSignature ... /> ?
1598     <sp:ProtectTokens ... /> ?
1599     <sp:OnlySignEntireHeadersAndBody ... /> ?
1600     ...
1601   </wsp:Policy>
1602   ...
1603 </sp:AsymmetricBinding>
  
```

1604
 1605 The following describes the attributes and elements listed in the schema outlined above:

1606 /sp: AsymmetricBinding
1607 This identifies a AsymmetricBinding assertion.

1608 /sp: AsymmetricBinding/wsp: Policy
1609 This indicates a nested `wsp: Policy` element that defines the behavior of the
1610 AsymmetricBinding assertion.

1611 /sp: AsymmetricBinding/wsp: Policy/sp: InitiatorToken
1612 This assertion indicates a requirement for an Initiator Token. The specified token populates the
1613 [Initiator Token] property and is used for the message signature from initiator to recipient, and
1614 encryption from recipient to initiator.

1615 /sp: AsymmetricBinding/wsp: Policy/sp: InitiatorToken/wsp: Policy
1616 The policy contained here MUST identify one or more token assertions.

1617 /sp: AsymmetricBinding/wsp: Policy/sp: RecipientToken
1618 This assertion indicates a requirement for a Recipient Token. The specified token populates the
1619 [Recipient Token] property and is used for encryption from initiator to recipient, and for the
1620 message signature from recipient to initiator.

1621 /sp: AsymmetricBinding/wsp: Policy/sp: RecipientToken/wsp: Policy
1622 The policy contained here MUST identify one or more token assertions.

1623 /sp: AsymmetricBinding/wsp: Policy/sp: AlgorithmSuite
1624 This assertion indicates a value that populates the [Algorithm Suite] property. See Section 8.1 for
1625 more details.

1626 /sp: AsymmetricBinding/wsp: Policy/sp: Layout
1627 This assertion indicates a value that populates the [Security Header Layout] property. See
1628 Section 8.2 for more details.

1629 /sp: AsymmetricBinding/wsp: Policy/sp: IncludeTimestamp
1630 This assertion indicates that the [Timestamp] property is set to 'true'.

1631 /sp: AsymmetricBinding/wsp: Policy/sp: EncryptBeforeSigning
1632 This assertion indicates that the [Protection Order] property is set to 'EncryptBeforeSigning'.

1633 /sp: AsymmetricBinding/wsp: Policy/sp: EncryptSignature
1634 This assertion indicates that the [Signature Protection] property is set to 'true'.

1635 /sp: AsymmetricBinding/wsp: Policy/sp: ProtectTokens
1636 This assertion indicates that the [Token Protection] property is set to 'true'.

1637 /sp: AsymmetricBinding/wsp: Policy/sp: OnlySignEntireHeadersAndBody
1638 This assertion indicates that the [Entire Header And Body Signatures] property is set to 'true'.

8 Supporting Tokens

1639

1640 Security Bindings use tokens to secure the message exchange. The Security Binding will
1641 require one to create a signature using the token identified in the Security Binding policy.
1642 This signature will here-to-fore be referred to as the “message signature”. Additional tokens
1643 may be specified to augment the claims provided by the token associated with the
1644 “message signature” provided by the Security Binding. This section defines four properties
1645 related to supporting token requirements which may be referenced by a Security Binding:
1646 [Supporting Tokens], [Signed Supporting Tokens], [Endorsing Supporting Tokens] and
1647 [Signed Endorsing Supporting Tokens]. Four assertions are defined to populate those
1648 properties: SupportingTokens, SignedSupportingTokens, EndorsingSupportingTokens, and
1649 SignedEndorsingSupportingTokens. These assertions SHOULD apply to [Endpoint Policy
1650 Subject]. These assertions MAY apply to [Message Policy Subject] or [Operation Policy
1651 Subject].

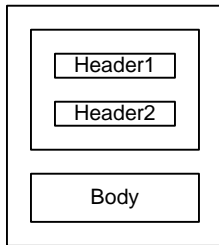
1652

1653 Supporting tokens may be specified at a different scope than the binding assertion which
1654 provides support for securing the exchange. For instance, a binding is specified at the scope
1655 of an endpoint, while the supporting tokens might be defined at the scope of a message.
1656 When assertions that populate this property are defined in overlapping scopes, the sender
1657 should merge the requirements by include all tokens from the outer scope and any
1658 additional tokens for a specific message from the inner scope.

1659

1660 To illustrate the different ways that supporting tokens may be bound to the message, let's
1661 consider a message with three components: Header1, Header2, and Body.

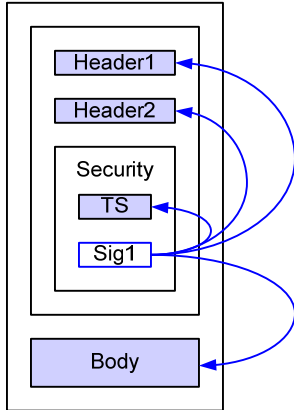
1662



1663

1664 Each binding requires that the message is signed using a token satisfying the required
1665 usage for that binding, and that the signature (Sig1) covers important parts of the message
1666 including the message timestamp (TS) facilitate replay detection. The signature is then
1667 included as part of the Security header as illustrated below:

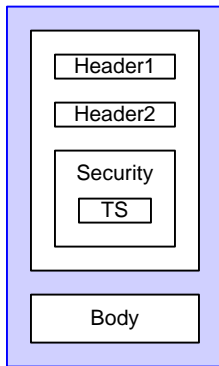
1668



1669

1670 Note: if required, the initiator may also include in the Security header the token used as the
 1671 basis for the message signature (Sig1), not shown in the diagram.

1672 If transport security is used, only the message timestamp (TS) is included in the Security
 1673 header as illustrated below:



1674

8.1 SupportingTokens Assertion

1676 Supporting tokens are included in the security header and may optionally include additional
 1677 message parts to sign and/or encrypt.

Syntax

```

1679 <sp:SupportingTokens ... >
1680   <wsp:Policy>
1681     [Token Assertion]+
1682     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite ?>
1683     (
1684       <sp:SignedParts ... > ... </sp:SignedParts> |
1685       <sp:SignedElements ... > ... </sp:SignedElements> |
1686       <sp:EncryptedParts ... > ... </sp:EncryptedParts> |
1687       <sp:EncryptedElements ... > ... </sp:EncryptedElements> |
1688     ) *
1689     ...
1690   </wsp:Policy>
1691   ...
1692 </sp:SupportingTokens>
  
```

1693

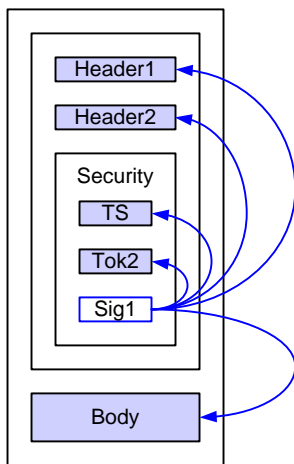
1694 The following describes the attributes and elements listed in the schema outlined above:

1695 /sp:SupportingTokens

1696 This identifies a SupportingTokens assertion. The specified tokens populate the
 1697 [Supporting Tokens] property.
 1698 /sp:SupportingTokens/wsp:Policy
 1699 This describes additional requirements for satisfying the SupportingTokens assertion.
 1700 /sp:SupportingTokens/wsp:Policy/[Token Assertion]
 1701 The policy MUST identify one or more token assertions.
 1702 /sp:SupportingTokens/wsp:Policy/sp:AlgorithmSuite
 1703 This optional element follows the schema outlined in Section 8.1 and describes the algorithms to
 1704 use for cryptographic operations performed with the tokens identified by this policy assertion.
 1705 /sp:SupportingTokens/wsp:Policy/sp:SignedParts
 1706 This optional element follows the schema outlined in Section 5.1.1 and describes additional
 1707 message parts that MUST be included in the signature generated with the token identified by this
 1708 policy assertion.
 1709 /sp:SupportingTokens/wsp:Policy/sp:SignedElements
 1710 This optional element follows the schema outlined in Section 5.1.2 and describes additional
 1711 message elements that MUST be included in the signature generated with the token identified by
 1712 this policy assertion.
 1713 /sp:SupportingTokens/wsp:Policy/sp:EncryptedParts
 1714 This optional element follows the schema outlined in Section 5.2.1 and describes additional
 1715 message parts that MUST be encrypted using the token identified by this policy assertion.
 1716 /sp:SupportingTokens/wsp:Policy/sp:EncryptedElements
 1717 This optional element follows the schema outlined in Section 5.2.1 and describes additional
 1718 message elements that MUST be encrypted using the token identified by this policy assertion.

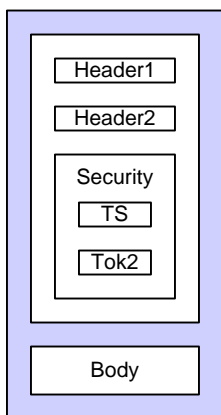
1719 **8.2 SignedSupportingTokens Assertion**

1720 Signed tokens are included in the "message signature" as defined above and may optionally
 1721 include additional message parts to sign and/or encrypt. The diagram below illustrates how
 1722 the attached token (Tok2) is signed by the message signature (Sig1):
 1723



1724
 1725 If transport security is used, the token (Tok2) is included in the Security header as
 1726 illustrated below:

1727



1728

1729 Syntax

1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743

```

<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>
  
```

1744

1745 The following describes the attributes and elements listed in the schema outlined above:

1746 /sp:SignedSupportingTokens

1747 This identifies a SignedSupportingTokens assertion. The specified tokens populate the
1748 [Signed Supporting Tokens] property.

1749 /sp:SignedSupportingTokens/wsp:Policy

1750 This describes additional requirements for satisfying the SignedSupportingTokens assertion.

1751 /sp:SignedSupportingTokens/wsp:Policy/[Token Assertion]

1752 The policy MUST identify one or more token assertions.

1753 /sp:SignedSupportingTokens/wsp:Policy/sp:AlgorithmSuite

1754 This optional element follows the schema outlined in Section 8.1 and describes the algorithms to
1755 use for cryptographic operations performed with the tokens identified by this policy assertion.

1756 /sp:SignedSupportingTokens/wsp:Policy/sp:SignedParts

1757 This optional element follows the schema outlined in Section 5.1.1 and describes additional
1758 message parts that MUST be included in the signature generated with the token identified by this
1759 policy assertion.

1760 /sp:SignedSupportingTokens/wsp:Policy/sp:SignedElements

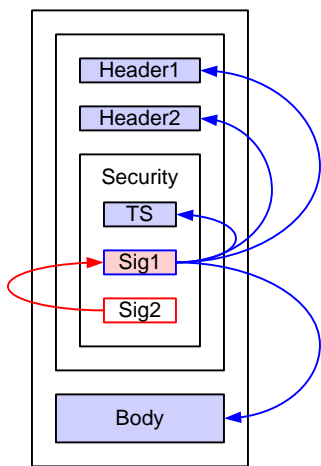
1761 This optional element follows the schema outlined in Section 5.1.2 and describes additional
1762 message elements that MUST be included in the signature generated with the token identified by
1763 this policy assertion.

1764 /sp:SignedSupportingTokens/wsp:Policy/sp:EncryptedParts
 1765 This optional element follows the schema outlined in Section 5.2.1 and describes additional
 1766 message parts that MUST be encrypted using the token identified by this policy assertion.

1767 /sp:SignedSupportingTokens/wsp:Policy/sp:EncryptedElements
 1768 This optional element follows the schema outlined in Section 5.2.1 and describes additional
 1769 message elements that MUST be encrypted using the token identified by this policy assertion.

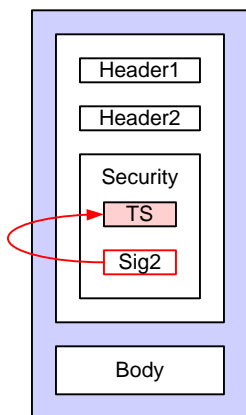
8.3 EndorsingSupportingTokens Assertion

1771 Endorsing tokens sign the message signature, that is they sign the entire `ds:Signature`
 1772 element produced from the message signature and may optionally include additional
 1773 message parts to sign and/or encrypt. The diagram below illustrates how the endorsing
 1774 signature (Sig2) signs the message signature (Sig1):



1776

1777 If transport security is used, the signature (Sig2) should cover the message timestamp as
 1778 illustrated below:



1780

1781 **Syntax**

```
<sp:EndorsingSupportingTokens ... >
  <wsp:Policy>
    [Token Assertion]+
    <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite> ?
  (
```

1787
1788
1789
1790
1791
1792
1793
1794
1795

```
<sp:SignedParts ... > ... </sp:SignedParts> |  
<sp:SignedElements ... > ... </sp:SignedElements> |  
<sp:EncryptedParts ... > ... </sp:EncryptedParts> |  
<sp:EncryptedElements ... > ... </sp:EncryptedElements> |  
  ) *  
  ...  
</wsp:Policy>  
  ...  
</sp:EndorsingSupportingTokens>
```

1796

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

1797

`/sp:EndorsingSupportingTokens`

1798

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

1800

`/sp:EndorsingSupportingTokens/wsp:Policy`

1801

This describes additional requirements for satisfying the `EndorsingSupportingTokens` assertion.

1802

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

1803

The policy MUST identify one or more token assertions.

1804

`/sp:EndorsingSupportingTokens/wsp:Policy/sp:AlgorithmSuite`

1805

This optional element follows the schema outlined in Section 8.1 and describes the algorithms to use for cryptographic operations performed with the tokens identified by this policy assertion.

1806

1807

`/sp:EndorsingSupportingTokens/wsp:Policy/sp:SignedParts`

1808

This optional element follows the schema outlined in Section 5.1.1 and describes additional message parts that MUST be included in the signature generated with the token identified by this policy assertion.

1809

1810

1811

`/sp:EndorsingSupportingTokens/wsp:Policy/sp:SignedElements`

1812

This optional element follows the schema outlined in Section 5.1.2 and describes additional message elements that MUST be included in the signature generated with the token identified by this policy assertion.

1813

1814

1815

`/sp:EndorsingSupportingTokens/wsp:Policy/sp:EncryptedParts`

1816

This optional element follows the schema outlined in Section 5.2.1 and describes additional message parts that MUST be encrypted using the token identified by this policy assertion.

1817

1818

`/sp:EndorsingSupportingTokens/wsp:Policy/sp:EncryptedElements`

1819

This optional element follows the schema outlined in Section 5.2.1 and describes additional message elements that MUST be encrypted using the token identified by this policy assertion.

1820

1821

8.4 SignedEndorsingSupportingTokens Assertion

1822

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

1823

1824

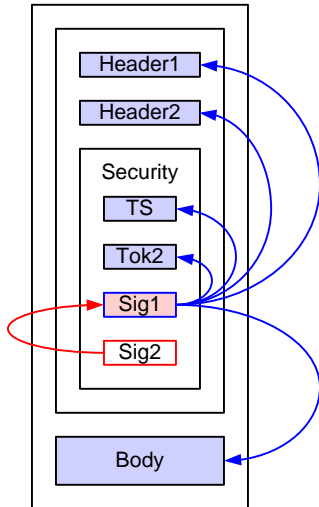
1825

1826

1827

1828

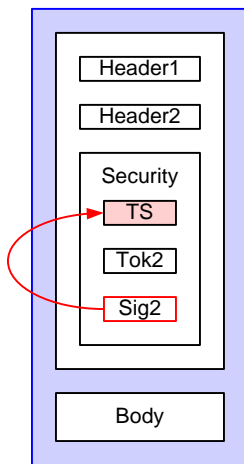
1829



1830

1831 If transport security is used, the token (Tok2) is included in the Security header and the
 1832 signature (Sig2) should cover the message timestamp as illustrated below:

1833



1834

1835 **Syntax**

```

1836 <sp:SignedEndorsingSupportingTokens ... >
1837   <wsp:Policy>
1838     [Token Assertion]+
1839     <sp:AlgorithmSuite ... > ... </sp:AlgorithmSuite ?
1840     (
1841       <sp:SignedParts ... > ... </sp:SignedParts> |
1842       <sp:SignedElements ... > ... </sp:SignedElements> |
1843       <sp:EncryptedParts ... > ... </sp:EncryptedParts> |
1844       <sp:EncryptedElements ... > ... </sp:EncryptedElements> |
1845     ) *
1846     ...
1847   </wsp:Policy>
1848   ...
1849 </sp:SignedEndorsingSupportingTokens>
  
```

1850

1851 The following describes the attributes and elements listed in the schema outlined above:

1852 /sp: SignedEndorsingSupportingTokens

1853 This identifies a SignedEndorsingSupportingTokens assertion. The specified tokens
1854 populate the [Signed Endorsing Supporting Tokens] property.
1855 /sp:SignedEndorsingSupportingTokens/wsp:Policy
1856 This describes additional requirements for satisfying the EndorsingSupportingTokens assertion.
1857 /sp:SignedEndorsingSupportingTokens/wsp:Policy/[Token Assertion]
1858 The policy MUST identify one or more token assertions.
1859 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:AlgorithmSuite
1860 This optional element follows the schema outlined in Section 8.1 and describes the algorithms to
1861 use for cryptographic operations performed with the tokens identified by this policy assertion.
1862 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:SignedParts
1863 This optional element follows the schema outlined in Section 5.1.1 and describes additional
1864 message parts that MUST be included in the signature generated with the token identified by this
1865 policy assertion.
1866 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:SignedElements
1867 This optional element follows the schema outlined in Section 5.1.2 and describes additional
1868 message elements that MUST be included in the signature generated with the token identified by
1869 this policy assertion.
1870 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:EncryptedParts
1871 This optional element follows the schema outlined in Section 5.2.1 and describes additional
1872 message parts that MUST be encrypted using the token identified by this policy assertion.
1873 /sp:SignedEndorsingSupportingTokens/wsp:Policy/sp:EncryptedElements
1874 This optional element follows the schema outlined in Section 5.2.1 and describes additional
1875 message elements that MUST be encrypted using the token identified by this policy assertion.

1876 8.5 Example

1877 Example policy containing supporting token assertions.

1878 `<!-- Example Endpoint Policy -->`


```

1879 <wsp:Policy>
1880   <sp:SymmetricBinding>
1881     <wsp:Policy>
1882       <sp:ProtectionToken>
1883         <sp:IssuedToken sp:IncludeToken=".../IncludeToken/Once" >
1884           <sp:Issuer>...</sp:Issuer>
1885           <sp:RequestSecurityTokenTemplate>
1886             ...
1887           </sp:RequestSecurityTokenTemplate>
1888         </sp:IssuedToken>
1889       </sp:ProtectionToken>
1890       <sp:AlgorithmSuite>
1891         <wsp:Policy>
1892           <sp:Basic256 />
1893         </wsp:Policy>
1894       </sp:AlgorithmSuite>
1895       ...
1896     <sp:SignedTokens>
1897       <wsp:Policy>
1898         <sp:UsernameToken sp:IncludeToken=".../IncludeToken/Once" />
1899       </wsp:Policy>
1900     </sp:SignedTokens>
1901     <sp:SignedEndorsingTokens>
1902       <wsp:Policy>
1903         <sp:X509V3Token sp:IncludeToken=".../IncludeToken/Once" />
1904       </wsp:Policy>
1905     </sp:SignedEndorsingTokens>
1906   </wsp:Policy>
1907 </sp:SymmetricBinding>
1908   ...
1909 </wsp:Policy>

```

1910 The sp:SignedTokens assertion in the above policy indicates that a Username Token must
1911 be included in the security header and covered by the message signature. The
1912 sp:SignedEndorsingTokens assertion indicates that an X509 certificate must be include in
1913 the security header and covered by the message signature. In addition, a signature over the
1914 message signature based on the key material associated with the X509 certificate must be
1915 include in the security header..

1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958

9 WSS: SOAP Message Security Options

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

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

Note: This approach is chosen because:

- A) [WSS: SOAP Message Security] allows for multiple equivalent reference mechanisms to be used in a single reference.
- B) In a multi-message exchange, a token may be referenced using different mechanisms depending on which of a series of messages is being secured.

WSS: SOAP Message Security 1.0 Properties

[Direct References]

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

[Key Identifier References]

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

[Issuer Serial References]

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

[External URI References]

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

1959 **[Embedded Token References]**

1960 This boolean property indicates whether the initiator and recipient MUST be able to process
1961 references that contain embedded tokens. A value of 'true' indicates that the initiator and
1962 recipient MUST be able to process such references. A value of 'false' indicates that the
1963 initiator and recipient MUST NOT generate such references and that the initiator and
1964 recipient MAY send a fault if such references are encountered. This property has a default
1965 value of 'false'.

1966

1967 **WSS: SOAP Message Security 1.1 Properties**

1968 **[Thumbprint References]**

1969 This boolean property indicates whether the initiator and recipient MUST be able to process
1970 references using token thumbprints. A value of 'true' indicates that the initiator and
1971 recipient MUST be able to process such references. A value of 'false' indicates that the
1972 initiator and recipient MUST NOT generate such references and that the initiator and
1973 recipient MAY send a fault if such references are encountered. This property has a default
1974 value of 'false'.

1975

1976 **[EncryptedKey References]**

1977 This boolean property indicates whether the initiator and recipient MUST be able to process
1978 references using EncryptedKey references. A value of 'true' indicates that the initiator and
1979 recipient MUST be able to process such references. A value of 'false' indicates that the
1980 initiator and recipient MUST NOT generate such references and that the initiator and
1981 recipient MAY send a fault if such references are encountered. This property has a default
1982 value of 'false'.

1983

1984 **[Signature Confirmation]**

1985 This boolean property specifies whether `wss11:SignatureConfirmation` elements should
1986 be used as defined in WSS: Soap Message Security 1.1. If the value is 'true',
1987 `wss11:SignatureConfirmation` elements MUST be used. If the value is 'false', signature
1988 confirmation elements MUST NOT be used. The value of this property applies to all
1989 signatures that are included in the security header. This property has a default value of
1990 'false'.

1991 **9.1 Wss10 Assertion**

1992 The Wss10 assertion allows you to specify which WSS: SOAP Message Security 1.0 options
1993 are supported.

1994 **Syntax**

```
1995 <sp:Wss10 ... >  
1996   <wsp:Policy>  
1997     <sp:MustSupportRefKeyIdentifier ... /> ?  
1998     <sp:MustSupportRefIssuerSerial ... /> ?  
1999     <sp:MustSupportRefExternalURI ... /> ?  
2000     <sp:MustSupportRefEmbeddedToken ... /> ?  
2001     ...  
2002   </wsp:Policy>  
2003   ...  
2004 </sp:Wss10>
```

2005

2006 The following describes the attributes and elements listed in the schema outlined above:

2007 /sp:Wss10

2008 This identifies a WSS10 assertion.

2009 /sp:Wss10/wsp:Policy

2010

2011 This indicates a policy that controls WSS: SOAP Message Security 1.0

2012 options./sp:Wss10/wsp:Policy/sp:MustSupportRefKeyIdentifier

2013 This assertion indicates that the [Key Identifier References] property is set to 'true'.

2014 /sp:Wss10/wsp:Policy/sp:MustSupportRefIssuerSerial

2015 This assertion indicates that the [Issuer Serial References] property is set to 'true'.

2016 /sp:Wss10/wsp:Policy/sp:MustSupportRefExternalURI

2017 This assertion indicates that the [External URI References] property is set to 'true'.

2018 /sp:Wss10/wsp:Policy/sp:MustSupportRefEmbeddedToken

2019 This assertion indicates that the [Embedded Token References] property is set to 'true'.

2020 9.2 Wss11 Assertion

2021 The Wss11 assertion allows you to specify which WSS: SOAP Message Security 1.1 options
2022 are supported.

2023 Syntax

```
2024 < sp:Wss11 ... >  
2025 <wsp:Policy>  
2026 <sp:MustSupportRefKeyIdentifier ... /> ?  
2027 <sp:MustSupportRefIssuerSerial ... /> ?  
2028 <sp:MustSupportRefExternalURI ... /> ?  
2029 <sp:MustSupportRefEmbeddedToken ... /> ?  
2030 <sp:MustSupportRefThumbprint ... /> ?  
2031 <sp:MustSupportRefEncryptedKey ... /> ?  
2032 <sp:RequireSignatureConfirmation ... /> ?  
2033 ...  
2034 </wsp:Policy>  
2035 </sp:Wss11>
```

2036

2037 The following describes the attributes and elements listed in the schema outlined above:

2038 /sp:Wss11

2039 This identifies an WSS11 assertion.

2040 /sp:Wss11/wsp:Policy

2041 This indicates a policy that controls WSS: SOAP Message Security 1.1 options.

2042 /sp:Wss11/wsp:Policy/sp:MustSupportRefKeyIdentifier

2043 This assertion indicates that the [Key Identifier References] property is set to 'true'.

2044 /sp:Wss11/wsp:Policy/sp:MustSupportRefIssuerSerial

2045 This assertion indicates that the [Issuer Serial References] property is set to 'true'.

2046 /sp:Wss11/wsp:Policy/sp:MustSupportRefExternalURI

2047 This assertion indicates that the [External URI References] property is set to 'true'.

2048 /sp:Wss11/wsp:Policy/sp:MustSupportRefEmbeddedToken

2049 This assertion indicates that the [Embedded Token References] property is set to 'true'.
2050 /sp:Wss11/wsp:Policy/sp:MustSupportRefThumbprint
2051 This assertion indicates that the [Thumbprint References] property is set to 'true'.
2052 /sp:Wss11/wsp:Policy/sp:MustSupportRefEncryptedKey
2053 This assertion indicates that the [EncryptedKey References] property is set to 'true'.
2054 /sp:Wss11/wsp:Policy/sp:RequireSignatureConfirmation
2055 This assertion indicates that the [Signature Confirmation] property is set to 'true'.

2056 10 WS-Trust Options

2057 This section defines the various policy assertions related to exchanges based on WS-Trust,
2058 specifically with client and server challenges and entropy behaviors. These assertions relate
2059 to interactions with a Security Token Service and may augment the behaviors defined by
2060 the Binding Property Assertions defined in Section 7. The assertions defined here MUST
2061 apply to [Endpoint Policy Subject].

2062

2063 WS-Trust 1.0 Properties

2064 [Client Challenge]

2065 This boolean property indicates whether client challenges are supported. A value of 'true'
2066 indicates that a `wst:SignChallenge` element is supported inside of an RST sent by the client
2067 to the server. A value of 'false' indicates that a `wst:SignChallenge` is not supported. There is
2068 no change in the number of messages exchanged by the client and service in satisfying the
2069 RST. This property has a default value of 'false'.

2070

2071 [Server Challenge]

2072 This boolean property indicates whether server challenges are supported. A value of 'true'
2073 indicates that a `wst:SignChallenge` element is supported inside of an RSTR sent by the
2074 server to the client. A value of 'false' indicates that a `wst:SignChallenge` is not supported. A
2075 challenge issued by the server may increase the number of messages exchanged by the
2076 client and service in order to accommodate the `wst:SignChallengeResponse` element sent by
2077 the client to the server in response to the `wst:SignChallenge` element. A final RSTR
2078 containing the issued token will follow subsequent to the server receiving the
2079 `wst:SignChallengeResponse` element. This property has a default value of 'false'.

2080

2081 [Client Entropy]

2082 This boolean property indicates whether client entropy is required to be used as key
2083 material for a requested proof token. A value of 'true' indicates that client entropy is
2084 required. A value of 'false' indicates that client entropy is not required. This property has a
2085 default value of 'false'.

2086

2087 [Server Entropy]

2088 This boolean property indicates whether server entropy is required to be used as key
2089 material for a requested proof token. A value of 'true' indicates that server entropy is
2090 required. A value of 'false' indicates that server entropy is not required. This property has a
2091 default value of 'false'.

2092 Note: If both the [Client Entropy] and [Server Entropy] properties are set to true, Client and
2093 server entropy are combined to produce a computed key using the Computed Key algorithm
2094 defined by the [Algorithm Suite] property.

2095

2096 [Issued Tokens]

2097 This boolean property indicates whether the `wst:IssuedTokens` header is supported as
2098 described in WS-Trust. A value of 'true' indicates that the `wst:IssuedTokens` header is

2099 supported. A value of 'false' indicates that the `wst:IssuedTokens` header is not supported.
2100 This property has a default value of 'false'.

2101 **10.1 Trust10 Assertion**

2102 The Trust10 assertion allows you to specify which WS-Trust 1.0 options are supported.

2103 **Syntax**

```
2104 <sp:Trust10 ... >  
2105 <wsp:Policy>  
2106 <sp:MustSupportClientChallenge ... />?  
2107 <sp:MustSupportServerChallenge ... />?  
2108 <sp:RequireClientEntropy ... />?  
2109 <sp:RequireServerEntropy ... />?  
2110 <sp:MustSupportIssuedTokens ... />?  
2111 ...  
2112 </wsp:Policy>  
2113 ...  
2114 </sp:Trust10 ... >
```

2115
2116 The following describes the attributes and elements listed in the schema outlined above:

2117 `/sp:Trust10`

2118 This identifies a Trust10 assertion.

2119 `/sp:Trust10/wsp:Policy`

2120 This indicates a policy that controls WS-Trust 1.0 options.

2121 `/sp:Trust10/wsp:Policy/sp:MustSupportClientChallenge`

2122 This assertion indicates that the [Client Challenge] property is set to 'true'.

2123 `/sp:Trust10/wsp:Policy/sp:MustSupportServerChallenge`

2124 This assertion indicates that the [Server Challenge] property is set to 'true'.

2125 `/sp:Trust10/wsp:Policy/sp:RequireClientEntropy`

2126 This assertion indicates that the [Client Entropy] property is set to 'true'.

2127 `/sp:Trust10/wsp:Policy/sp:RequireServerEntropy`

2128 This assertion indicates that the [Server Entropy] property is set to 'true'.

2129 `/sp:Trust10/wsp:Policy/sp:MustSupportIssuedTokens`

2130 This assertion indicates that the [Issued Tokens] property is set to 'true'.

2131 **11 Security Considerations**

2132 It is strongly recommended that policies and assertions be signed to prevent tampering.
2133 It is recommended that policies should not be accepted unless they are signed and have an
2134 associated security token to specify the signer has proper claims for the given policy. That
2135 is, a party shouldn't rely on a policy unless the policy is signed and presented with sufficient
2136 claims. It is further recommended that the entire policy exchange mechanism be protected
2137 to prevent man-in-the-middle downgrade attacks.

2138
2139 It should be noted that the mechanisms described in this document could be secured as
2140 part of a SOAP message using [WSS: SOAP Message Security](#) or embedded within other
2141 objects using object-specific security mechanisms.

2142
2143 It is recommended that policies not specify two (or more) SignedSupportingTokens or
2144 SignedEndorsingSupportingTokens of the same token type. Messages conforming to such
2145 policies are subject to modification which may be undetectable.

2146
2147 It is recommended that policies specify the OnlySignEntireHeadersAndBody assertion along
2148 with the rest of the policy in order to combat certain XML substitution attacks.

2149 **A. Assertions and WS-PolicyAttachment**

2150 This non-normative appendix classifies assertions according to their suggested scope in
2151 WSDL 1.1 per Section 4 of [WS-PolicyAttachment]. See Figure 1 in Section 4.1 of [WS-
2152 PolicyAttachment] for a graphical representation of the relationship between policy scope
2153 and WSDL.

2154 **A.1 Endpoint Policy Subject Assertions**

2155 **A.1.1 Security Binding Assertions**

2156 TransportBinding Assertion (Section 8.3)
2157 SymmetricBinding Assertion (Section 8.4)
2158 AsymmetricBinding Assertion (Section 8.5)

2159 **A.1.2 Token Assertions**

2160 SupportingTokens Assertion (Section 9.1)
2161 SignedSupportingTokens Assertion (Section 9.2)
2162 EndorsingSupportingTokens Assertion (Section 9.3)
2163 SignedEndorsingSupportingTokens Assertion (Section 9.4)

2164 **A.1.3 WSS: SOAP Message Security 1.0 Assertions**

2165 Wss10 Assertion (Section 10.1)

2166 **A.1.4 WSS: SOAP Message Security 1.1 Assertions**

2167 Wss11 Assertion (Section 10.2)

2168 **A.1.5 Trust 1.0 Assertions**

2169 Trust10 Assertion (Section 11.1)

2170 **A.2 Operation Policy Subject Assertions**

2171 **A.2.1 Supporting Token Assertions**

2172 SupportingTokens Assertion (Section 9.1)
2173 SignedSupportingTokens Assertion (Section 9.2)
2174 EndorsingSupportingTokens Assertion (Section 9.3)
2175 SignedEndorsingSupportingTokens Assertion (Section 9.4)

2176 **A.3 Message Policy Subject Assertions**

2177 **A.3.1 Supporting Token Assertions**

2178 SupportingTokens Assertion (Section 9.1)
2179 SignedSupportingTokens Assertion (Section 9.2)

2180	EndorsingSupportingTokens Assertion	(Section 9.3)
2181	SignedEndorsingSupportingTokens Assertion	(Section 9.4)
2182	A.3.2 Protection Assertions	
2183	SignedParts Assertion	(Section 5.1.1)
2184	SignedElements Assertion	(Section 5.1.2)
2185	EncryptedParts Assertion	(Section 5.2.1)
2186	EncryptedElements Assertion	(Section 5.2.2)
2187	RequiredElements Assertion	(Section 5.3.1)
2188	A.4 Assertions With Undefined Policy Subject	
2189	The assertions listed in this section do not have a defined policy subject because they appear nested	
2190	inside some other assertion which does have a defined policy subject.	
2191	A.4.1 General Assertions	
2192	AlgorithmSuite Assertion	(Section 8.1)
2193	Layout Assertion	(Section 8.2)
2194	IncludeTimestamp Assertion	(Section 8.3)
2195	IncludeTimestamp Assertion	(Section 8.4)
2196	EncryptBeforeSigning Assertion	(Section 8.4)
2197	EncryptSignature Assertion	(Section 8.4)
2198	ProtectTokens Assertion	(Section 8.4)
2199	OnlySignEntireHeadersAndBody Assertion	(Section 8.4)
2200	IncludeTimestamp Assertion	(Section 8.5)
2201	EncryptBeforeSigning Assertion	(Section 8.5)
2202	EncryptSignature Assertion	(Section 8.5)
2203	ProtectTokens Assertion	(Section 8.5)
2204	OnlySignEntireHeadersAndBody Assertion	(Section 8.5)
2205	A.4.2 Token Usage Assertions	
2206	TransportToken Assertion	(Section 8.3)
2207	EncryptionToken Assertion	(Section 8.4)
2208	SignatureToken Assertion	(Section 8.4)
2209	ProtectionToken Assertion	(Section 8.4)
2210	InitiatorToken Assertion	(Section 8.5)
2211	RecipientToken Assertion	(Section 8.5)
2212	A.4.3 Token Assertions	
2213	UsernameToken Assertion	(Section 6.3.1)
2214	IssuedToken Assertion	(Section 6.3.2)
2215	X509Token Assertion	(Section 6.3.3)
2216	KerberosToken Assertion	(Section 6.3.4)

2217	SpnegoContextToken Assertion	(Section 6.3.5)
2218	SecurityContextToken Assertion	(Section 6.3.6)
2219	SecureConversationToken Assertion	(Section 6.3.7)
2220	SamlToken Assertion	(Section 6.3.8)
2221	RelToken Assertion	(Section 6.3.9)
2222	HttpsToken Assertion	(Section 6.3.10)

2223 **A.4.4 WSS: SOAP Message Security 1.0 Assertions**

2224	MustSupportRefKeyIdentifier Assertion	(Section 10.1)
2225	MustSupportRefIssuerSerial Assertion	(Section 10.1)
2226	MustSupportRefExternalUri Assertion	(Section 10.1)
2227	MustSupportRefEmbeddedToken Assertion	(Section 10.1)

2228 **A.4.5 WSS: SOAP Message Security 1.1 Assertions**

2229	MustSupportRefKeyIdentifier Assertion	(Section 10.2)
2230	MustSupportRefIssuerSerial Assertion	(Section 10.2)
2231	MustSupportRefExternalUri Assertion	(Section 10.2)
2232	MustSupportRefEmbeddedToken Assertion	(Section 10.2)
2233	MustSupportRefThumbprint Assertion	(Section 10.2)
2234	RequireSignatureConfirmation Assertion	(Section 10.2)

2235 **A.4.6 Trust 1.0 Assertions**

2236	MustSupportClientChallenge Assertion	(Section 11.1)
2237	MustSupportServerChallenge Assertion	(Section 11.1)
2238	RequireClientEntropy Assertion	(Section 11.1)
2239	RequireServerEntropy Assertion	(Section 11.1)
2240	MustSupportIssuedTokens Assertion	(Section 11.1)

2241

B. Issued Token Policy

2242
2243

The section provides further detail about behavior associated with the IssuedToken assertion in section 6.2.2.

2244

2245

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

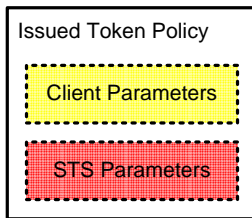
2250

2251

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

2252

2253



2254

2255

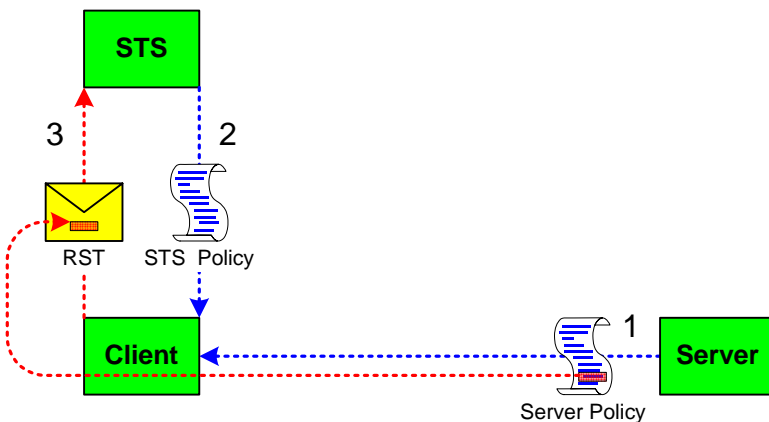
The client-specific parameters of the Issued Token policy assertion along with the remainder of the server policy are consumed by the client. The STS specific parameters of the Issued Token policy assertion are passed on to the STS by copying the parameters directly into the RST request sent by the Client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

2256

2257

2258

2259



2260

2261

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

2262

2263

2264

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

2265

2266

2267

2268 The Issued Token Policy Assertion contains elements which must be understood by the Client. The
2269 assertion contains one element which contains a list of arbitrary elements which should be sent along to
2270 the STS by copying the elements as-is directly into the request sent by the Client to the STS following the
2271 protocol defined in WS-Trust.
2272
2273 Elements inside the `sp:RequestSecurityTokenTemplate` element MUST conform to WS-Trust [WS-
2274 Trust]. All items are optional, since the Server and STS may already have a pre-arranged relationship
2275 which specifies some or all of the conditions and constraints for issued tokens.

2276

C. Strict Security Header Layout Examples

2277 The following sections describe the security header layout for specific bindings when
2278 applying the 'Strict' layout rules defined in Section 7.7.

2279 C.1 Transport Binding

2280 This section describes how the 'Strict' security header layout rules apply to the Transport
2281 Binding.

2282 C.1.1 Policy

2283 The following example shows a policy indicating a Transport Binding, an Https Token as the
2284 Transport Token, an algorithm suite, a requirement to include tokens in the supporting
2285 signatures, a username token attached to the message, and finally an X509 token attached
2286 to the message and endorsing the message signature. No message protection requirements
2287 are described since the transport covers all message parts.

```
2288 <wsp:Policy>
2289   <sp:TransportBinding>
2290     <wsp:Policy>
2291       <sp:TransportToken>
2292         <wsp:Policy>
2293           <sp:HttpsToken />
2294         </wsp:Policy>
2295       </sp:TransportToken>
2296       <sp:AlgorithmSuite>
2297         <wsp:Policy>
2298           <sp:Basic256 />
2299         </wsp:Policy>
2300       </sp:AlgorithmSuite>
2301       <sp:Layout>
2302         <wsp:Policy>
2303           <sp:Strict />
2304         </wsp:Policy>
2305       </sp:Layout>
2306       <sp:IncludeTimestamp />
2307       <sp:SignedSupportingTokens>
2308         <wsp:Policy>
2309           <sp:UsernameToken sp:IncludeToken="../../../IncludeToken/Once" />
2310         </wsp:Policy>
2311       </sp:SignedSupportingTokens>
2312       <sp:SignedEndorsingSupportingTokens>
2313         <wsp:Policy>
2314           <sp:X509V3Token sp:IncludeToken="../../../IncludeToken/Once" />
2315         </wsp:Policy>
2316       </sp:SignedEndorsingSupportingTokens>
2317     </wsp:Policy>
2318   </sp:TransportBinding>
2319   <sp:Wss11>
2320     <sp:RequireSignatureConfirmation />
2321   </sp:Wss11>
2322 </wsp:Policy>
```

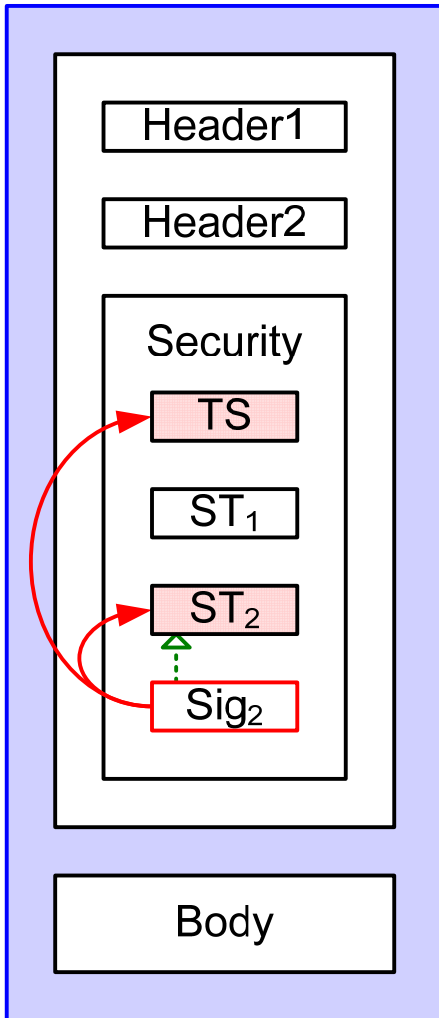
2323 This policy is used as the basis for the examples shown in the subsequent section describing
2324 the security header layout for this binding.

2325 C.1.2 Initiator to Recipient Messages

2326 Messages sent from initiator to recipient have the following layout for the security header:

- 2327 1. A `wsu:Timestamp` element.
- 2328 2. Any tokens contained in the [Signed Supporting Tokens] property.
- 2329 3. Any tokens contained in the [Signed Endorsing Supporting Tokens] property each
2330 followed by the corresponding signature. Each signature MUST cover the
2331 `wsu:Timestamp` element from 1 above and SHOULD cover any other unique identifier
2332 for the message in order to prevent replays. If [Token Protection] is 'true', the
2333 signature MUST also cover the supporting token. If [Derived Keys] is 'true' and the
2334 supporting token is associated with a symmetric key, then a Derived Key Token,
2335 based on the supporting token, appears between the supporting token and the
2336 signature.
- 2337 4. Any signatures for tokens contained in the [Endorsing Supporting Tokens] property.
2338 Each signature MUST cover the `wsu:Timestamp` element from 1 above and SHOULD
2339 cover at least some other unique identifier for the message in order to prevent
2340 replays. If [Token Protection] is 'true', the signature MUST also cover the supporting
2341 token. If [Derived Keys] is 'true' and the supporting token is associated with a
2342 symmetric key, then a Derived Key Token, based on the supporting token, appears
2343 before the signature.

2344 The following diagram illustrates the security header layout for the initiator to recipient
2345 message:



2346

2347 The blue outer box shows that the entire message is protected (signed and encrypted) by
 2348 the transport. The red arrows (left) from the box labeled Sig₂ indicate the parts signed by
 2349 the supporting token labeled ST₂, namely the message timestamp labeled TS and the token
 2350 used as the basis for the signature labeled ST₂. The green dotted arrow indicates the token
 2351 that was used as the basis for the signature. In general, the ordering of the items in the
 2352 security header follows the most optimal layout for a receiver to process its contents.

2353 *Example:*

2354 Initiator to recipient message


```

2355 <S:Envelope>
2356   <S:Header>
2357     ...
2358     <wsse:Security>
2359       <wsu:Timestamp wsu:Id="timestamp">
2360         <wsu:Created>[datetime]</wsu:Created>
2361         <wsu:Expires>[datetime]</wsu:Expires>
2362       </wsu:Timestamp>
2363       <wsse:UsernameToken wsu:Id='SomeSignedToken' >
2364         ...
2365       </wsse:UsernameToken>
2366       <wsse:BinarySecurityToken wsu:Id="SomeSignedEndorsingToken" >
2367         ...
2368       </wsse:BinarySecurityToken>
2369       <ds:Signature>
2370         <ds:SignedInfo>
2371           <ds:References>
2372             <ds:Reference URI="#timestamp" />
2373             <ds:Reference URI="#SomeSignedEndorsingToken" />
2374           </ds:References>
2375         </ds:SignedInfo>
2376         <ds:Signature>...</ds:Signature>
2377         <ds:KeyInfo>
2378           <wsse:SecurityTokenReference>
2379             <wsse:Reference URI="#SomeSignedEndorsingToken" />
2380           </wsse:SecurityTokenReference>
2381         </ds:KeyInfo>
2382       </ds:Signature>
2383     ...
2384   </wsse:Security>
2385   ...
2386 </S:Header>
2387 <S:Body>
2388   ...
2389 </S:Body>
2390 </S:Envelope>

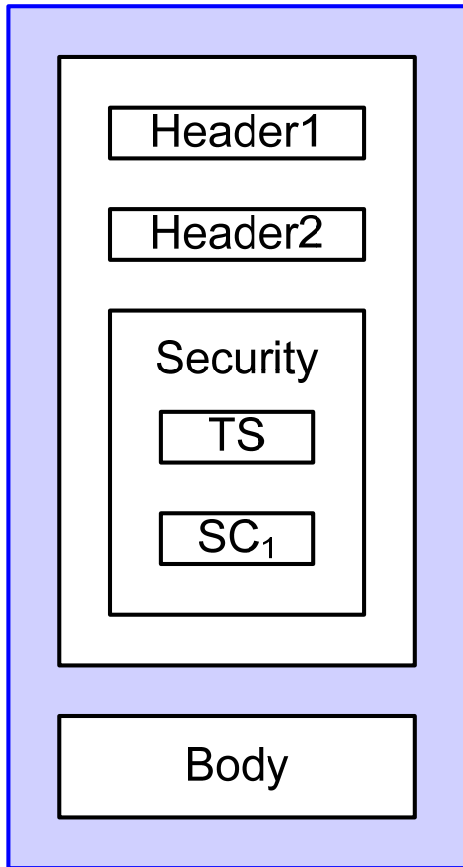
```

2391 C.1.3 Recipient to Initiator Messages

2392 Messages sent from recipient to initiator have the following layout for the security header:

- 2393 1. A `wsu:Timestamp` element.
- 2394 2. If the [Signature Confirmation] property has a value of 'true', then a
2395 `wsse11:SignatureConfirmation` element for each signature in the corresponding
2396 message sent from initiator to recipient. If there are no signatures in the
2397 corresponding message from the initiator to the recipient, then a
2398 `wsse11:SignatureConfirmation` element with no Value attribute.

2399 The following diagram illustrates the security header layout for the recipient to initiator
2400 message:



2401
 2402 The blue outer box shows that the entire message is protected (signed and encrypted) by
 2403 the transport. One `wsse11:SignatureConfirmation` element labeled `SC1` corresponding to
 2404 the signature in the initial message illustrated previously is included. In general, the
 2405 ordering of the items in the security header follows the most optimal layout for a receiver to
 2406 process its contents.

2407 *Example:*

2408 Recipient to initiator message

```

2409 <S:Envelope>
2410 <S:Header>
2411   ...
2412 <wsse:Security>
2413   <wsu:Timestamp wsu:Id="timestamp">
2414     <wsu:Created>[datetime]</wsu:Created>
2415     <wsu:Expires>[datetime]</wsu:Expires>
2416   </wsu:Timestamp>
2417   <wsse11:SignatureConfirmation Value="..." />
2418   ...
2419 </wsse:Security>
2420   ...
2421 </S:Header>
2422 <S:Body>
2423   ...
2424 </S:Body>
2425 </S:Envelope>
  
```

2426 **C.2 Symmetric Binding**

2427 This section describes how the 'Strict' security header layout rules apply to the Symmetric
2428 Binding.

2429 **C.2.1 Policy**

2430 The following example shows a policy indicating a Symmetric Binding, a symmetric key
2431 based IssuedToken provided as the Protection Token, an algorithm suite, a requirement to
2432 encrypt the message parts before signing, a requirement to encrypt the message signature,
2433 a requirement to include tokens in the message signature and the supporting signatures, a
2434 username token attached to the message, and finally an X509 token attached to the
2435 message and endorsing the message signature. Minimum message protection requirements
2436 are described as well.

```

2437 <!-- Example Endpoint Policy -->
2438 <wsp:Policy>
2439   <sp:SymmetricBinding>
2440     <wsp:Policy>
2441       <sp:ProtectionToken>
2442         <sp:IssuedToken sp:IncludeToken="../../../IncludeToken/Once" >
2443           <sp:Issuer>...</sp:Issuer>
2444           <sp:RequestSecurityTokenTemplate>
2445             ...
2446           </sp:RequestSecurityTokenTemplate>
2447         </sp:IssuedToken>
2448       </sp:ProtectionToken>
2449       <sp:AlgorithmSuite>
2450         <wsp:Policy>
2451           <sp:Basic256 />
2452         </wsp:Policy>
2453       </sp:AlgorithmSuite>
2454       <sp:Layout>
2455         <wsp:Policy>
2456           <sp:Strict />
2457         </wsp:Policy>
2458       </sp:Layout>
2459       <sp:IncludeTimestamp />
2460       <sp:EncryptBeforeSigning />
2461       <sp:EncryptSignature />
2462       <sp:ProtectTokens />
2463       <sp:SignedSupportingTokens>
2464         <wsp:Policy>
2465           <sp:UsernameToken sp:IncludeToken="../../../IncludeToken/Once" />
2466         </wsp:Policy>
2467       </sp:SignedSupportingTokens>
2468       <sp:SignedEndorsingSupportingTokens>
2469         <wsp:Policy>
2470           <sp:X509V3Token sp:IncludeToken="../../../IncludeToken/Once" />
2471         </wsp:Policy>
2472       </sp:SignedEndorsingSupportingTokens>
2473     </wsp:Policy>
2474   </sp:SymmetricBinding>
2475   <sp:Wss11>
2476     <wsp:Policy>
2477       <sp:RequireSignatureConfirmation />
2478     </wsp:Policy>
2479   </sp:Wss11>
2480 </wsp:Policy>
2481
2482 <!-- Example Message Policy -->
2483 <wsp:Policy>
2484   <sp:SignedParts>
2485     <sp:Header Name="Header1" Namespace="..." />
2486     <sp:Header Name="Header2" Namespace="..." />
2487     <sp:Body/>
2488   </sp:SignedParts>
2489   <sp:EncryptedParts>
2490     <sp:Header Name="Header2" Namespace="..." />
2491     <sp:Body/>
2492   </sp:EncryptedParts>
2493 </wsp:Policy>

```

2494 This policy is used as the basis for the examples shown in the subsequent section describing
2495 the security header layout for this binding.

2496 C.2.2 Initiator to Recipient Messages

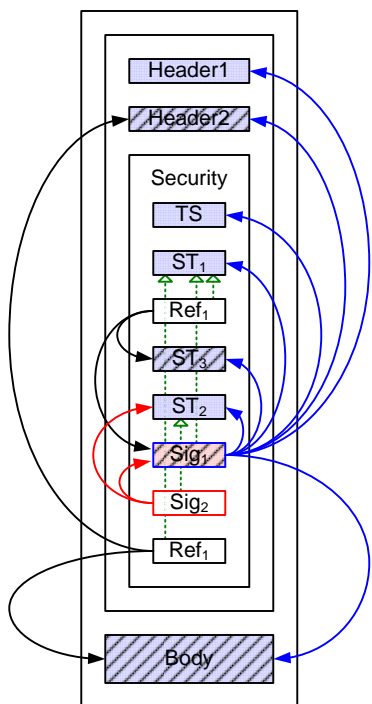
2497 Messages sent from initiator to recipient have the following layout for the security header:

- 2498 1. A `wsu:Timestamp` element if `[Timestamp]` is 'true'.
- 2499 2. If the `sp:IncludeToken` attribute on the `[Encryption Token]` is `.../IncludeToken/Once`
2500 or `.../IncludeToken/Always`, then the `[Encryption Token]`.
- 2501 3. If `[Derived Keys]` is 'true', then a Derived Key Token, based on the `[Encryption`
2502 `Token]`. This Derived Key Token is used for encryption.
- 2503 4. A reference list including references to encrypted items. If `[Signature Protection]` is
2504 'true', then the reference list MUST include a reference to the message signature. If
2505 `[Protection Order]` is 'SignBeforeEncrypting', then the reference list MUST include a
2506 reference to all the message parts specified in the `EncryptedParts` assertions in the
2507 policy. If `[Derived Keys]` is 'true', then the key in the token from 3 above MUST be
2508 used, otherwise the key in the `[Encryption Token]`.
- 2509 5. Any tokens from the `[Signed Supporting Tokens]` and `[Signed Endorsing Supporting`
2510 `Tokens]` properties whose `sp:IncludeToken` attribute is `.../IncludeToken/Once` or
2511 `.../IncludeToken/Always`.
- 2512 6. If the `[Signature Token]` is not the same as the `[Encryption Token]`, and the
2513 `sp:IncludeToken` attribute on the `[Signature Token]` is `.../IncludeToken/Once` or
2514 `.../IncludeToken/Always`, then the `[Signature Token]`.
- 2515 7. If `[Derived Keys]` is 'true', then a Derived Key Token based on the `[Signature`
2516 `Token]`. This Derived Key Token is used for signature.
- 2517 8. A signature over the `wsu:Timestamp` from 1 above, any tokens from 5 above
2518 regardless of whether they are included in the message, and any message parts
2519 specified in `SignedParts` assertions in the policy. If `[Token Protection]` is 'true', the
2520 signature MUST cover the `[Signature Token]` regardless of whether it is included in
2521 the message. If `[Derived Keys]` is 'true', the key in the token from 7 above MUST be
2522 used, otherwise the key in the `[Signature Token]` from 6 above.
- 2523 9. Signatures covering the main signature from 8 above for any tokens from the
2524 `[Endorsing Supporting Tokens]` and `[Signed Endorsing Supporting Tokens]`
2525 properties. If `[Token Protection]` is 'true', the signature MUST also cover the
2526 endorsing token. If `[Derived Keys]` is 'true' and the endorsing token is associated
2527 with a symmetric key, then a Derived Key Token, based on the endorsing token,
2528 appears before the signature.
- 2529 10. If `[Protection Order]` is 'EncryptBeforeSigning', then a reference list referencing all
2530 the message parts specified in `EncryptedParts` assertions in the policy. If `[Derived`
2531 `Keys]` is 'true', then the key in the token from 3 above MUST be used, otherwise the
2532 key in the `[Encryption Token]` from 2 above.

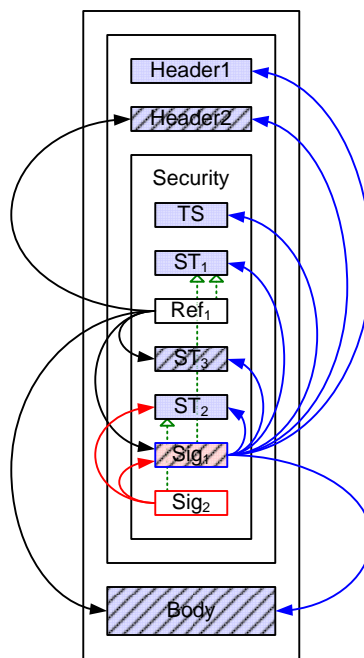
2533

2534 The following diagram illustrates the security header layout for the initiator to recipient
2535 message:

Encrypt Then Sign



Sign Then Encrypt



2536

2537 The blue arrows (right) indicate parts that were signed as part of the message signature
2538 labeled Sig₁. The red arrows (left) from the box labeled Sig₂ indicate the parts signed by the
2539 supporting token labeled ST₂, namely the message signature labeled Sig₁ and the token
2540 used as the basis for the signature labeled ST₂. The black arrows (left) from boxes labeled
2541 Ref₁ indicate references to parts encrypted using a key based on the Shared Secret Token
2542 labeled ST₁. The green dotted arrows indicate the token that was used as the basis for each
2543 cryptographic operation. In general, the ordering of the items in the security header follows
2544 the most optimal layout for a receiver to process its contents.

2545 *Example:*

2546 Initiator to recipient message using EncryptBeforeSigning.

```

2547 <S:Envelope>
2548   <S:Header>
2549     <x:Header1 wsu:Id="Header1" >
2550       ...
2551     </x:Header1>
2552     <wsse1:EncryptedHeader wsu:Id="enc_Header2">
2553       <!-- Plaintext Header2
2554       <x:Header2 wsu:Id="Header2" >
2555         ...
2556       </x:Header2>
2557       -->
2558       ...
2559     </wsse1:EncryptedHeader>
2560     ...
2561     <wsse:Security>
2562       <wsu:Timestamp wsu:Id="Timestamp">
2563         <wsu:Created>...</wsu:Created>
2564         <wsu:Expires>...</wsu:Expires>
2565       </wsu:Timestamp>
2566       <saml:Assertion AssertionId="_SharedSecretToken" ...>
2567         ...
2568       </saml:Assertion>
2569       <xenc:ReferenceList>
2570         <xenc:DataReference URI="#enc_Signature" />
2571         <xenc:DataReference URI="#enc_SomeUsernameToken" />
2572         ...
2573       </xenc:ReferenceList>
2574       <xenc:EncryptedData ID="enc_SomeUsernameToken" >
2575         <!-- Plaintext UsernameToken
2576         <wsse:UsernameToken wsu:Id="SomeUsernameToken" >
2577           ...
2578         </wsse:UsernameToken>
2579         -->
2580         ...
2581         <ds:KeyInfo>
2582           <wsse:SecurityTokenReference>
2583             <wsse:Reference URI="#_SharedSecretToken" />
2584           </wsse:SecurityTokenReference>
2585         </ds:KeyInfo>
2586       </xenc:EncryptedData>
2587       <wsse:BinarySecurityToken wsu:Id="SomeSupportingToken" >
2588         ...
2589       </wsse:BinarySecurityToken>
2590       <xenc:EncryptedData ID="enc_Signature">
2591         <!-- Plaintext Signature
2592         <ds:Signature Id="Signature">
2593           <ds:SignedInfo>
2594             <ds:References>
2595               <ds:Reference URI="#Timestamp" >...</ds:Reference>
2596               <ds:Reference URI="#SomeUsernameToken" >...</ds:Reference>
2597               <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
2598               <ds:Reference URI="#_SharedSecretToken" >...</ds:Reference>
2599               <ds:Reference URI="#Header1" >...</ds:Reference>
2600               <ds:Reference URI="#Header2" >...</ds:Reference>
2601               <ds:Reference URI="#Body" >...</ds:Reference>
2602             </ds:References>
2603           </ds:SignedInfo>
2604           <ds:Signature>...</ds:Signature>
2605           <ds:KeyInfo>
2606             <wsse:SecurityTokenReference>
2607               <wsse:Reference URI="#_SharedSecretToken" />
2608             </wsse:SecurityTokenReference>
2609           </ds:KeyInfo>
2610         </ds:Signature>

```

```

2611 -->
2612 ...
2613 <ds:KeyInfo>
2614   <wsse:SecurityTokenReference>
2615     <wsse:Reference URI="#_SharedSecretToken" />
2616   </wsse:SecurityTokenReference>
2617 </ds:KeyInfo>
2618 </xenc:EncryptedData>
2619 <ds:Signature>
2620   <ds:SignedInfo>
2621     <ds:References>
2622       <ds:Reference URI="#Signature" >...</ds:Reference>
2623       <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
2624     </ds:References>
2625   </ds:SignedInfo>
2626   <ds:Signature>...</ds:Signature>
2627 </ds:KeyInfo>
2628   <wsse:SecurityTokenReference>
2629     <wsse:Reference URI="#SomeSupportingToken" />
2630   </wsse:SecurityTokenReference>
2631 </ds:KeyInfo>
2632 </ds:Signature>
2633 <xenc:ReferenceList>
2634   <xenc:DataReference URI="#enc_Body" />
2635   <xenc:DataReference URI="#enc_Header2" />
2636   ...
2637 </xenc:ReferenceList>
2638 </wsse:Security>
2639 </S:Header>
2640 <S:Body>
2641   <xenc:EncryptedData Id="enc_Body">
2642     ...
2643     <ds:KeyInfo>
2644       <wsse:SecurityTokenReference>
2645         <wsse:Reference URI="#_SharedSecretToken" />
2646       </wsse:SecurityTokenReference>
2647     </ds:KeyInfo>
2648   </xenc:EncryptedData>
2649 </S:Body>
2650 </S:Envelope>

```

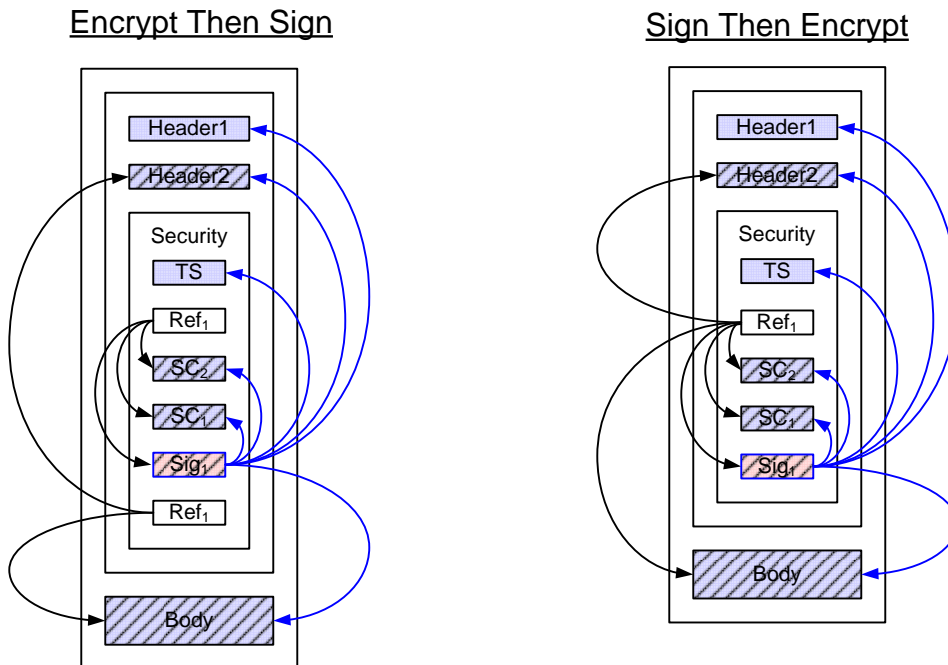
2651 C.2.3 Recipient to Initiator Messages

2652 Messages send from recipient to initiator have the following layout for the security header:

- 2653 1. A `wsu:Timestamp` element if [Timestamp] is 'true'.
- 2654 2. If the `sp:IncludeToken` attribute on the [Encryption Token] is
2655 `.../IncludeToken/Always`, then the [Encryption Token].
- 2656 3. If [Derived Keys] is 'true', then a Derived Key Token, based on the [Encryption
2657 Token]. This Derived Key Token is used for encryption.
- 2658 4. A reference list including references to encrypted items. If [Signature Protection] is
2659 'true', then the reference list MUST include a reference to the message signature
2660 from 6 below, and the `wssell:SignatureConfirmation` elements from 5 below if
2661 any. If [Protection Order] is 'SignBeforeEncrypting', then the reference list MUST
2662 include a reference to all the message parts specified in the EncryptedParts
2663 assertions in the policy. If [Derived Keys] is 'true', then the key in the token from 2
2664 above MUST be used, otherwise the key in the [Encryption Token] from 2 above.
- 2665 5. If [Signature Confirmation] is 'true' then a `wssell:SignatureConfirmation` element
2666 for each signature in the corresponding message sent from initiator to recipient. If

- 2667 there are no signatures in the corresponding message from the initiator to the
 2668 recipient, then a `wssell:SignatureConfirmation` element with no Value attribute.
- 2669 6. If the [Signature Token] is not the same as the [Encryption Token], and the
 2670 `sp:IncludeToken` attribute on the [Signature Token] is `.../IncludeToken/Always`,
 2671 then the [Signature Token].
- 2672 7. If [Derived Keys] is 'true', then a Derived Key Token, based on the [Signature
 2673 Token]. This Derived Key Token is used for signature.
- 2674 8. A signature over the `wsu:Timestamp` from 1 above, any
 2675 `wssell:SignatureConfirmation` elements from 5 above, and all the message parts
 2676 specified in SignedParts assertions in the policy. If [Token Protection] is 'true', the
 2677 signature MUST also cover the [Signature Token] regardless of whether it is included
 2678 in the message. If [Derived Keys] is 'true', the key in the token from 6 above MUST
 2679 be used, otherwise the key in the [Signature Token].
- 2680 9. If [Protection Order] is 'EncryptBeforeSigning' then a reference list referencing all the
 2681 message parts specified in EncryptedParts assertions in the policy. If [Derived Keys]
 2682 is 'true', then the key in the Derived Key Token from 3 above MUST be used,
 2683 otherwise the key in the [Encryption Token].

2684 The following diagram illustrates the security header layout for the recipient to initiator
 2685 message:



2686

2687 The blue arrows (right) indicate parts that were signed as part of the message signature
 2688 labeled `Sig1`. The black arrows (left) from boxes labeled `Ref1` indicate references to parts
 2689 encrypted using a key based on the [SharedSecret Token] (not shown in these diagrams as
 2690 it is referenced as an external token). Two `wssell:SignatureConfirmation` elements
 2691 labeled `SC1` and `SC2` corresponding to the two signatures in the initial message illustrated
 2692 previously is included. In general, the ordering of the items in the security header follows
 2693 the most optimal layout for a receiver to process its contents. The rules used to determine
 2694 this ordering are described in Appendix C.

2695 *Example:*

2696 Recipient to initiator message using EncryptBeforeSigning.

```

2697 <S:Envelope>
2698   <S:Header>
2699     <x:Header1 wsu:Id="Header1" >
2700       ...
2701     </x:Header1>
2702     <wssell:EncryptedHeader wsu:Id="enc_Header2">
2703       <!-- Plaintext Header2
2704       <x:Header2 wsu:Id="Header2" >
2705         ...
2706       </x:Header2>
2707       -->
2708       ...
2709     </wssell:EncryptedHeader>
2710     ...
2711     <wsse:Security>
2712       <wsu:Timestamp wsu:Id="Timestamp">
2713         <wsu:Created>...</wsu:Created>
2714         <wsu:Expires>...</wsu:Expires>
2715       </wsu:Timestamp>
2716       <xenc:ReferenceList>
2717         <xenc:DataReference URI="#enc_Signature" />
2718         <xenc:DataReference URI="#enc_SigConf1" />
2719         <xenc:DataReference URI="#enc_SigConf2" />
2720         ...
2721       </xenc:ReferenceList>
2722       <xenc:EncryptedData ID="enc_SigConf1" >
2723         <!-- Plaintext SignatureConfirmation
2724         <wssell:SignatureConfirmation wsu:Id="SigConf1" >
2725           ...
2726         </wssell:SignatureConfirmation>
2727         -->
2728         ...
2729       </xenc:EncryptedData>
2730       <xenc:EncryptedData ID="enc_SigConf2" >
2731         <!-- Plaintext SignatureConfirmation
2732         <wssell:SignatureConfirmation wsu:Id="SigConf2" >
2733           ...
2734         </wssell:SignatureConfirmation>
2735         -->
2736         ...
2737       </xenc:EncryptedData>
2738       <xenc:EncryptedData Id="enc_Signature">
2739         <!-- Plaintext Signature
2740         <ds:Signature Id="Signature">
2741           <ds:SignedInfo>
2742             <ds:References>
2743               <ds:Reference URI="#Timestamp" >...</ds:Reference>
2744               <ds:Reference URI="#SigConf1" >...</ds:Reference>
2745               <ds:Reference URI="#SigConf2" >...</ds:Reference>
2746               <ds:Reference URI="#Header1" >...</ds:Reference>
2747               <ds:Reference URI="#Header2" >...</ds:Reference>
2748               <ds:Reference URI="#Body" >...</ds:Reference>
2749             </ds:References>
2750           </ds:SignedInfo>
2751           <ds:Signature>...</ds:Signature>
2752           <ds:KeyInfo>
2753             <wsse:SecurityTokenReference>
2754               <wsse:Reference URI="#_SomeIssuedToken" />
2755             </wsse:SecurityTokenReference>
2756           </ds:KeyInfo>
2757         </ds:Signature>
2758         -->
2759         ...
2760       </ds:Signature>

```

```

2761     <wsse:SecurityTokenReference>
2762         <wsse:Reference URI="#_SomeIssuedToken" />
2763     </wsse:SecurityTokenReference>
2764 </ds:KeyInfo>
2765 <xenc:EncryptedData>
2766 <xenc:ReferenceList>
2767     <xenc:DataReference URI="#enc_Body" />
2768     <xenc:DataReference URI="#enc_Header2" />
2769     ...
2770 </xenc:ReferenceList>
2771 </wsse:Security>
2772 </S:Header>
2773 <S:Body>
2774 <xenc:EncryptedData Id="enc_Body">
2775     ...
2776 <ds:KeyInfo>
2777     <wsse:SecurityTokenReference>
2778         <wsse:Reference URI="#_SomeIssuedToken" />
2779     </wsse:SecurityTokenReference>
2780 </ds:KeyInfo>
2781 </xenc:EncryptedData>
2782 </S:Body>
2783 </S:Envelope>

```

2784 C.3 Asymmetric Binding

2785 This section describes how the 'Strict' security header layout rules apply to the Asymmetric
2786 Binding.

2787 C.3.1 Policy

2788 The following example shows a policy indicating an Asymmetric Binding, an X509 token as
2789 the [Initiator Token], an X509 token as the [Recipient Token], an algorithm suite, a
2790 requirement to encrypt the message parts before signing, a requirement to encrypt the
2791 message signature, a requirement to include tokens in the message signature and the
2792 supporting signatures, a requirement to include `wsse11:SignatureConfirmation` elements,
2793 a username token attached to the message, and finally an X509 token attached to the
2794 message and endorsing the message signature. Minimum message protection requirements
2795 are described as well.

```

2796 <!-- Example Endpoint Policy -->
2797 <wsp:Policy>
2798   <sp:AsymmetricBinding>
2799     <wsp:Policy>
2800       <sp:RecipientToken>
2801         <wsp:Policy>
2802           <sp:X509V3Token sp:IncludeToken=".../IncludeToken/Always" />
2803         </wsp:Policy>
2804       </sp:RecipientToken>
2805       <sp:InitiatorToken>
2806         <wsp:Policy>
2807           <sp:X509V3Token sp:IncludeToken=".../IncludeToken/Always" />
2808         </wsp:Policy>
2809       </sp:InitiatorToken>
2810       <sp:AlgorithmSuite>
2811         <wsp:Policy>
2812           <sp:Basic256 />
2813         </wsp:Policy>
2814       </sp:AlgorithmSuite>
2815       <sp:Layout>
2816         <wsp:Policy>
2817           <sp:Strict />
2818         </wsp:Policy>
2819       </sp:Layout>
2820       <sp:IncludeTimestamp />
2821       <sp:EncryptBeforeSigning />
2822       <sp:EncryptSignature />
2823       <sp:ProtectTokens />
2824       <sp:SignedSupportingTokens>
2825         <wsp:Policy>
2826           <sp:UsernameToken sp:IncludeToken=".../IncludeToken/Once" />
2827         </wsp:Policy>
2828       </sp:SignedSupportingTokens>
2829       <sp:SignedEndorsingSupportingTokens>
2830         <wsp:Policy>
2831           <sp:X509V3Token sp:IncludeToken=".../IncludeToken/Once" />
2832         </wsp:Policy>
2833       </sp:SignedEndorsingSupportingTokens>
2834     </wsp:Policy>
2835   </sp:AsymmetricBinding>
2836   <sp:Wss11>
2837     <wsp:Policy>
2838       <sp:RequireSignatureConfirmation />
2839     </wsp:Policy>
2840   </sp:Wss11>
2841 </wsp:Policy>
2842
2843 <!-- Example Message Policy -->
2844 <wsp>All>
2845   <sp:SignedParts>
2846     <sp:Header Name="Header1" Namespace="..." />
2847     <sp:Header Name="Header2" Namespace="..." />
2848     <sp:Body/>
2849   </sp:SignedParts>
2850   <sp:EncryptedParts>
2851     <sp:Header Name="Header2" Namespace="..." />
2852     <sp:Body/>
2853   </sp:EncryptedParts>
2854 </wsp>All>

```

2855
2856 This policy is used as the basis for the examples shown in the subsequent section describing
2857 the security header layout for this binding.

2858 C.3.2 Initiator to Recipient Messages

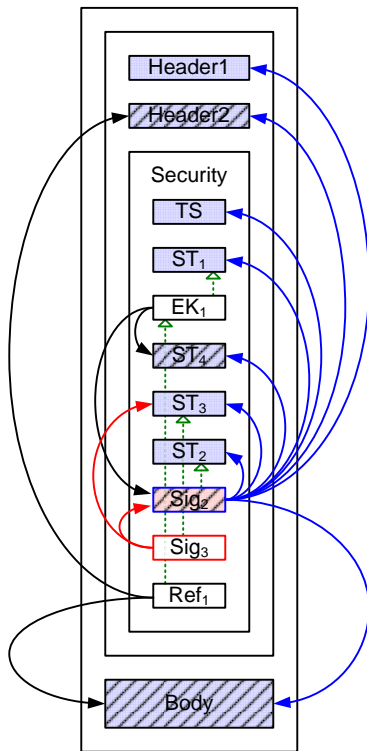
2859 Messages sent from initiator to recipient have the following layout:

- 2860 1. A `wsu:Timestamp` element if [Timestamp] is 'true'.
- 2861 2. If a [Recipient Token] is specified, and the associated `sp:IncludeToken` attribute is
2862 `.../IncludeToken/Once` or `.../IncludeToken/Always`, then the [Recipient Token].
- 2863 3. If a [Recipient Token] is specified then an `xenc:EncryptedKey` element, containing a
2864 key encrypted for the recipient. If [Protection Order] is 'SignBeforeEncrypting' then
2865 the `xenc:EncryptedKey` element MUST include an `xenc:ReferenceList` containing a
2866 reference to all the message parts specified in `EncryptedParts` assertions in the
2867 policy. If [Signature Protection] is 'true' then the reference list MUST contain a
2868 reference to the message signature from 6 below. It is an error if [Signature
2869 Protection] is 'true' and there is not a message signature.
- 2870 4. Any tokens from the [Signed Supporting Tokens] and [Signed Endorsing Supporting
2871 Tokens] properties whose `sp:IncludeToken` attribute is `.../IncludeToken/Once` or
2872 `.../IncludeToken/Always`.
- 2873 5. If an [Initiator Token] is specified, and the associated `sp:IncludeToken` attribute is
2874 `.../IncludeToken/Once` or `.../IncludeToken/Always`, then the [Initiator Token].
- 2875 6. A signature based on the key in the [Initiator Token] if specified, over the
2876 `wsu:Timestamp` from 1 above, any tokens from 4 above regardless of whether they
2877 are included in the message, and any message parts specified in `SignedParts`
2878 assertions in the policy. If [Token Protection] is 'true', the signature MUST also cover
2879 the [Initiator Token] regardless of whether it is included in the message.
- 2880 7. Signatures for tokens from the [Endorsing Supporting Tokens] and [Signed
2881 Endorsing Supporting Tokens] properties. If [Derived Keys] is 'true' and the
2882 supporting token is associated with a symmetric key, then a Derived Key Token,
2883 based on the supporting token, appears before the signature. If [Token Protection] is
2884 'true', the signature MUST also cover the supporting token regardless of whether it is
2885 included in the message.
- 2886 8. If a [Recipient Token] is specified and [Protection Order] is 'EncryptBeforeSigning'
2887 then a reference list including a reference to all the message parts specified in
2888 `EncryptedParts` assertions in the policy. The encrypted parts MUST reference the key
2889 contained in the `xenc:EncryptedKey` element from 3 above.

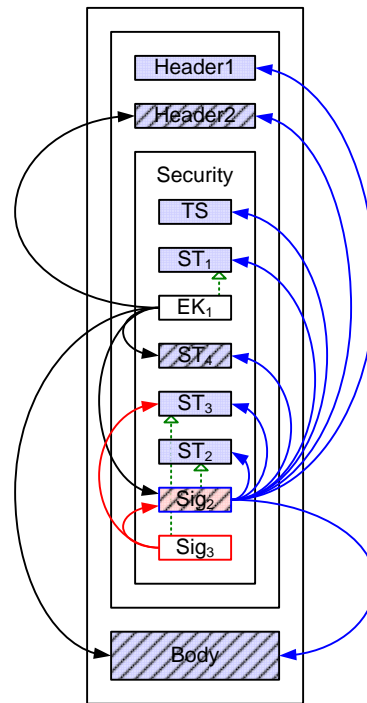
2890

2891 The following diagram illustrates the security header layout for the initiator to recipient
2892 messages:

Encrypt Then Sign



Sign Then Encrypt



2893

2894 The blue arrows (right) indicate parts that were signed as part of the message signature
2895 labeled Sig₂ using the [Initiator Token] labeled ST₂. The red arrows (left) from the box
2896 labeled Sig₃ indicate the parts signed by the supporting token ST₃, namely the message
2897 signature Sig₂ and the token used as the basis for the signature labeled ST₃. The black
2898 arrows (left) from boxes labeled EK₁ indicate references to parts encrypted using a key
2899 encrypted for the [Recipient Token] labeled ST₁. The black arrows (left) from boxes labeled
2900 Ref₁ indicate additional references to parts encrypted using the key contained in the
2901 encrypted key labeled EK₁. The green dotted arrows indicate the token used as the basis for
2902 each cryptographic operation. In general, the ordering of the items in the security header
2903 follows the most optimal layout for a receiver to process its contents. The rules used to
2904 determine this ordering are described in Appendix C.

2905

2906 Note: In most typical scenarios, the recipient key is not included in the message, but rather
2907 the encrypted key contains an external reference to the token containing the encryption
2908 key. The diagram illustrates how one might attach a security token related to the encrypted
2909 key for completeness. One possible use-case for this approach might be a stack which does
2910 not support the STR Dereferencing Transform, but wishes to include the encryption token in
2911 the message signature.

2912 Initiator to recipient message *Example*

2913

```

2914 <S:Envelope>
2915   <S:Header>
2916     <x:Header1 wsu:Id="Header1" >
2917       ...
2918     </x:Header1>
2919     <wsse1:EncryptedHeader wsu:Id="enc_Header2">
2920       <!-- Plaintext Header2
2921       <x:Header2 wsu:Id="Header2" >
2922         ...
2923       </x:Header2>
2924       -->
2925     </wsse1:EncryptedHeader>
2926     ...
2927   <wsse:Security>
2928     <wsu:Timestamp wsu:Id="Timestamp">
2929       <wsu:Created>...</wsu:Created>
2930       <wsu:Expires>...</wsu:Expires>
2931     </wsu:Timestamp>
2932     <wsse:BinarySecurityToken wsu:Id="RecipientToken" >
2933       ...
2934     </wsse:BinarySecurityToken>
2935     <xenc:EncryptedKey wsu:Id="RecipientEncryptedKey" >
2936       ...
2937     <xenc:ReferenceList>
2938       <xenc:DataReference URI="#enc_Signature" />
2939       <xenc:DataReference URI="#enc_SomeUsernameToken" />
2940       ...
2941     </xenc:ReferenceList>
2942   </xenc:EncryptedKey>
2943   <xenc:EncryptedData ID="enc_SomeUsernameToken" >
2944     <!-- Plaintext UsernameToken
2945     <wsse:UsernameToken wsu:Id="SomeUsernameToken" >
2946       ...
2947     </wsse:UsernameToken>
2948     -->
2949     ...
2950   </xenc:EncryptedData>
2951   <wsse:BinarySecurityToken wsu:Id="SomeSupportingToken" >
2952     ...
2953   </wsse:BinarySecurityToken>
2954   <wsse:BinarySecurityToken wsu:Id="InitiatorToken" >
2955     ...
2956   </wsse:BinarySecurityToken>
2957   <xenc:EncryptedData ID="enc_Signature">
2958     <!-- Plaintext Signature
2959     <ds:Signature Id="Signature">
2960       <ds:SignedInfo>
2961         <ds:References>
2962           <ds:Reference URI="#Timestamp" >...</ds:Reference>
2963           <ds:Reference URI="#SomeUsernameToken" >...</ds:Reference>
2964           <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
2965           <ds:Reference URI="#RecipientToken" >...</ds:Reference>
2966           <ds:Reference URI="#InitiatorToken" >...</ds:Reference>
2967           <ds:Reference URI="#Header1" >...</ds:Reference>
2968           <ds:Reference URI="#Header2" >...</ds:Reference>
2969           <ds:Reference URI="#Body" >...</ds:Reference>
2970         </ds:References>
2971       </ds:SignedInfo>
2972     </ds:Signature>...</ds:Signature>
2973     <ds:KeyInfo>
2974       <wsse:SecurityTokenReference>
2975         <wsse:Reference URI="#InitiatorToken" />
2976       </wsse:SecurityTokenReference>
2977

```



```

2978     </ds:KeyInfo>
2979     </ds:Signature>
2980     -->
2981     ...
2982 </xenc:EncryptedData>
2983 <ds:Signature>
2984   <ds:SignedInfo>
2985     <ds:References>
2986       <ds:Reference URI="#Signature" >...</ds:Reference>
2987       <ds:Reference URI="#SomeSupportingToken" >...</ds:Reference>
2988     </ds:References>
2989   </ds:SignedInfo>
2990 </ds:Signature>...</ds:Signature>
2991 <ds:KeyInfo>
2992   <wsse:SecurityTokenReference>
2993     <wsse:Reference URI="#SomeSupportingToken" />
2994   </wsse:SecurityTokenReference>
2995 </ds:KeyInfo>
2996 </ds:Signature>
2997 <xenc:ReferenceList>
2998   <xenc:DataReference URI="#enc_Body" />
2999   <xenc:DataReference URI="#enc_Header2" />
3000   ...
3001 </xenc:ReferenceList>
3002 </wsse:Security>
3003 </S:Header>
3004 <S:Body>
3005   <xenc:EncryptedData Id="enc_Body">
3006     ...
3007     <ds:KeyInfo>
3008       <wsse:SecurityTokenReference>
3009         <wsse:Reference URI="#RecipientEncryptedKey" />
3010       </wsse:SecurityTokenReference>
3011     </ds:KeyInfo>
3012   </xenc:EncryptedData>
3013 </S:Body>
3014 </S:Envelope>

```

3015 C.3.3 Recipient to Initiator Messages

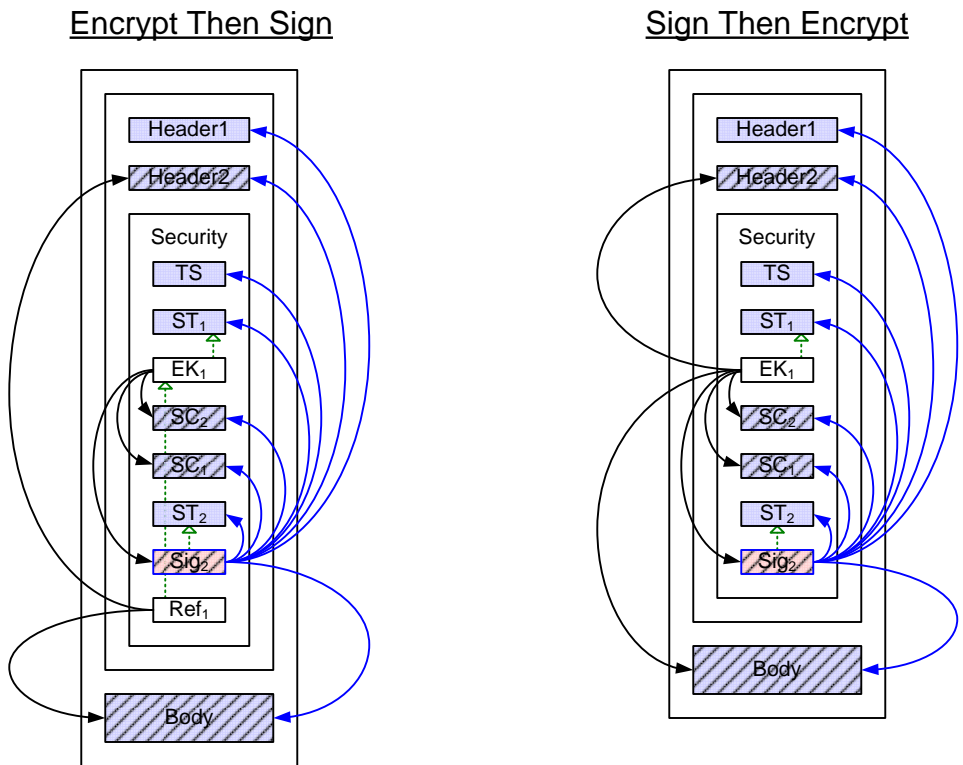
3016 Messages sent from recipient to initiator have the following layout:

- 3017 1. A `wsu:Timestamp` element if `[Timestamp]` is 'true'.
- 3018 2. If an `[Initiator Token]` is specified, and the associated `sp:IncludeToken` attribute is
3019 `.../IncludeToken/Always`, then the `[Initiator Token]`.
- 3020 3. If an `[Initiator Token]` is specified then an `xenc:EncryptedKey` element, containing a
3021 key encrypted for the initiator. If `[Protection Order]` is 'SignBeforeEncrypting' then
3022 the `xenc:EncryptedKey` element MUST include an `xenc:ReferenceList` containing a
3023 reference to all the message parts specified in `EncryptedParts` assertions in the
3024 policy. If `[Signature Protection]` is 'true' then the reference list MUST also contain a
3025 reference to the message signature from 6 below, if any and references to the
3026 `wsse11:SignatureConfirmation` elements from 4 below, if any.
- 3027 4. If `[Signature Confirmation]` is 'true', then a `wsse11:SignatureConfirmation`
3028 element for each signature in the corresponding message sent from initiator to
3029 recipient. If there are no signatures in the corresponding message from the initiator
3030 to the recipient, then a `wsse11:SignatureConfirmation` element with no `Value`
3031 attribute.

- 3032 5. If a [Recipient Token] is specified, and the associated `sp:IncludeToken` attribute is
 3033 `.../IncludeToken/Always`, then the [Recipient Token].
- 3034 6. If a [Recipient Token] is specified, then a signature based on the key in the
 3035 [Recipient Token], over the `wsu:Timestamp` from 1 above, the
 3036 `wsse11:SignatureConfirmation` elements from 4 above, and any message parts
 3037 specified in `SignedParts` assertions in the policy. If [Token Protection] is 'true' and
 3038 the [Initiator Token] is specified, then the signature MUST also cover the [Initiator
 3039 Token].
- 3040 7. If an [Initiator Token] is specified and [Protection Order] is 'EncryptBeforeSigning'
 3041 then a reference list including a reference to all the message parts specified in
 3042 `EncryptedParts` assertions in the policy. The encrypted parts MUST reference the key
 3043 contained in the `xenc:EncryptedKey` element from 3 above.

3044

3045 The following diagram illustrates the security header layout for the recipient to initiator
 3046 messages:



3047

3048 The blue arrows (right) indicate parts that were signed as part of the message signature
 3049 labeled Sig₂ using the [Recipient Token] labeled ST₂. The black arrows (left) from boxes
 3050 labeled EK₁ indicate references to parts encrypted using a key encrypted for the [Recipient
 3051 Token] labeled ST₁. The black arrows (left) from boxes labeled Ref₁ indicate additional
 3052 references to parts encrypted using the key contained in the encrypted key labeled EK₁. The
 3053 green dotted arrows indicate the token used as the basis for each cryptographic operation.
 3054 Two `wsse11:SignatureConfirmation` elements labeled SC₁ and SC₂ corresponding to the
 3055 two signatures in the initial message illustrated previously is included. In general, the
 3056 ordering of the items in the security header follows the most optimal layout for a receiver to
 3057 process its contents. The rules used to determine this ordering are described in Appendix C.
 3058 Recipient to initiator message *Example*:

```

3059 <S:Envelope>
3060 <S:Header>
3061 <x:Header1 wsu:Id="Header1" >
3062 ...
3063 </x:Header1>
3064 <wssell:EncryptedHeader wsu:Id="enc_Header2">
3065 <!-- Plaintext Header2
3066 <x:Header2 wsu:Id="Header2" >
3067 ...
3068 </x:Header2>
3069 -->
3070 ...
3071 </wssell:EncryptedHeader>
3072 ...
3073 <wsse:Security>
3074 <wsu:Timestamp wsu:Id="Timestamp">
3075 <wsu:Created>...</wsu:Created>
3076 <wsu:Expires>...</wsu:Expires>
3077 </wsu:Timestamp>
3078 <wsse:BinarySecurityToken wsu:Id="InitiatorToken" >
3079 ...
3080 </wsse:BinarySecurityToken>
3081 <xenc:EncryptedKey wsu:Id="InitiatorEncryptedKey" >
3082 ...
3083 <xenc:ReferenceList>
3084 <xenc:DataReference URI="#enc_Signature" />
3085 <xenc:DataReference URI="#enc_SigConf1" />
3086 <xenc:DataReference URI="#enc_SigConf2" />
3087 ...
3088 </xenc:ReferenceList>
3089 </xenc:EncryptedKey>
3090 <xenc:EncryptedData ID="enc_SigConf2" >
3091 <!-- Plaintext SignatureConfirmation
3092 <wssell:SignatureConfirmation wsu:Id="SigConf2" ...>
3093 ...
3094 </wssell:SignatureConfirmation>
3095 -->
3096 ...
3097 </xenc:EncryptedData>
3098 <xenc:EncryptedData ID="enc_SigConf1" >
3099 <!-- Plaintext SignatureConfirmation
3100 <wssell:SignatureConfirmation wsu:Id="SigConf1" ...>
3101 ...
3102 </wssell:SignatureConfirmation>
3103 -->
3104 ...
3105 </xenc:EncryptedData>
3106 <wsse:BinarySecurityToken wsu:Id="RecipientToken" >
3107 ...
3108 </wsse:BinarySecurityToken>
3109 <xenc:EncryptedData ID="enc_Signature">
3110 <!-- Plaintext Signature
3111 <ds:Signature Id="Signature">
3112 <ds:SignedInfo>
3113 <ds:References>
3114 <ds:Reference URI="#Timestamp" >...</ds:Reference>
3115 <ds:Reference URI="#SigConf1" >...</ds:Reference>
3116 <ds:Reference URI="#SigConf2" >...</ds:Reference>
3117 <ds:Reference URI="#RecipientToken" >...</ds:Reference>
3118 <ds:Reference URI="#InitiatorToken" >...</ds:Reference>
3119 <ds:Reference URI="#Header1" >...</ds:Reference>
3120 <ds:Reference URI="#Header2" >...</ds:Reference>
3121 <ds:Reference URI="#Body" >...</ds:Reference>
3122 </ds:References>

```

```
3123     </ds:SignedInfo>
3124     <ds:Signature>...</ds:Signature>
3125     <ds:KeyInfo>
3126         <wsse:SecurityTokenReference>
3127             <wsse:Reference URI="#RecipientToken" />
3128         </wsse:SecurityTokenReference>
3129     </ds:KeyInfo>
3130 </ds:Signature>
3131 -->
3132 ...
3133 </xenc:EncryptedData>
3134 <xenc:ReferenceList>
3135     <xenc:DataReference URI="#enc_Body" />
3136     <xenc:DataReference URI="#enc_Header2" />
3137     ...
3138 </xenc:ReferenceList>
3139 </wsse:Security>
3140 </S:Header>
3141 <S:Body>
3142     <xenc:EncryptedData Id="enc_Body">
3143         ...
3144         <ds:KeyInfo>
3145             <wsse:SecurityTokenReference>
3146                 <wsse:Reference URI="#InitiatorEncryptedKey" />
3147             </wsse:SecurityTokenReference>
3148         </ds:KeyInfo>
3149     </xenc:EncryptedData>
3150 </S:Body>
3151 </S:Envelope>
```

3152

D. Acknowledgements

3153 The following individuals have participated in the creation of this specification and are gratefully
3154 acknowledged:

3155 **Original Authors:**

3156 Giovanni Della-Libera, Microsoft

3157 Martin Gudgin, Microsoft

3158 Phillip Hallam-Baker, VeriSign

3159 Maryann Hondo, IBM

3160 Hans Granqvist, Verisign

3161 Chris Kaler, Microsoft (editor)

3162 Hiroshi Maruyama, IBM

3163 Michael McIntosh, IBM

3164 Anthony Nadalin, IBM (editor)

3165 Nataraj Nagaratnam, IBM

3166 Rob Philpott, RSA Security

3167 Hemma Prafullchandra, VeriSign

3168 John Shewchuk, Microsoft

3169 Doug Walter, Microsoft

3170 Riaz Zolfonoon, RSA Security

3171

3172 **Original Acknowledgements:**

3173 Vaithialingam B. Balayoghan, Microsoft

3174 Francisco Curbera, IBM

3175 Christopher Ferris, IBM

3176 Cedric Fournet, Microsoft

3177 Andy Gordon, Microsoft

3178 Tomasz Janczuk, Microsoft

3179 David Melgar, IBM

3180 Bruce Rich, IBM

3181 Jeffrey Schlimmer, Microsoft

3182 Chris Sharp, IBM

3183 Kent Tamura, IBM

3184 T.R. Vishwanath, Microsoft

3185 Elliot Waingold, Microsoft

E. Non-Normative Text

3187

F. Revision History

3188

[optional; should not be included in OASIS Standards]

3189

Revision	Date	Editor	Changes Made
0.1	12-06-2005	Anthony Nadalin	Initial Conversion to OASIS Template

3190

3191