



Web Services Security

X.509 Certificate Token Profile 1.1

OASIS Standard incorporating Approved Errata,
01 November 2006

OASIS Identifier:

wss-v1.1-spec-errata-os-X509TokenProfile

Document Location:

<http://docs.oasis-open.org/wss/v1.1/>

Technical Committee:

Web Service Security (WSS)

Chairs:

Kelvin Lawrence, IBM
Chris Kaler, Microsoft

Editors:

Anthony Nadalin, IBM
Chris Kaler, Microsoft
Ronald Monzillo, Sun
Phillip Hallam-Baker, Verisign

Abstract:

This document describes how to use X.509 Certificates with the Web Services Security: SOAP Message Security specification [WS-Security] specification.

Status:

This is an OASIS Standard document produced by the Web Services Security Technical Committee. It was approved by the OASIS membership on 1 February 2006. Check the current location noted above for possible errata to this document.

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

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to

36
37

the Intellectual Property Rights section of the WS-Security TC web page
(<http://www.oasis-open.org/committees/wss/ipr.php>).

38 **Notices**

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

51

52 Copyright (C) OASIS Open 2002-2006. All Rights Reserved.

53

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

63

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

66

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

72

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

75

76 This section is non-normative.

77 **Table of Contents**

78 1 Introduction (Non-Normative) 5
79 2 Notations and Terminology (Normative) 6
80 2.1 Notational Conventions 6
81 2.2 Namespaces 6
82 2.3 Terminology 7
83 3 Usage (Normative) 8
84 3.1 Token types 8
85 3.1.1 X509v3 Token Type 8
86 3.1.2 X509PKIPathv1 Token Type 8
87 3.1.3 PKCS7 Token Type 8
88 3.2 Token References 9
89 3.2.1 Reference to an X.509 Subject Key Identifier 9
90 3.2.2 Reference to a Security Token 10
91 3.2.3 Reference to an Issuer and Serial Number 10
92 3.3 Signature 10
93 3.3.1 Key Identifier 10
94 3.3.2 Reference to a Binary Security Token 12
95 3.3.3 Reference to an Issuer and Serial Number 13
96 3.4 Encryption 13
97 3.5 Error Codes 15
98 4 Threat Model and Countermeasures (Non-Normative) 16
99 5 References 17
100 Appendix A: Acknowledgments 19
101 Appendix B: Revision History 22
102

103 **1 Introduction (Non-Normative)**

104 This specification describes the use of the X.509 authentication framework with the Web Services
105 Security: SOAP Message Security specification [WS-Security].

106

107 An X.509 certificate specifies a binding between a public key and a set of attributes that includes
108 (at least) a subject name, issuer name, serial number and validity interval. This binding may be
109 subject to subsequent revocation advertised by mechanisms that include issuance of CRLs,
110 OCSP tokens or mechanisms that are outside the X.509 framework, such as XKMS.

111

112 An X.509 certificate may be used to validate a public key that may be used to authenticate a
113 SOAP message or to identify the public key with a SOAP message that has been encrypted.

114

115 Note that Sections 2.1, 2.2, all of 3, and indicated parts of 5 are normative. All other sections are
116 non-normative.

117 2 Notations and Terminology (Normative)

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

119 2.1 Notational Conventions

120 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",
121 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be
122 interpreted as described in RFC 2119.

123

124 When describing abstract data models, this specification uses the notational convention used by
125 the XML Infoset. Specifically, abstract property names always appear in square brackets (e.g.,
126 [some property]).

127

128 When describing concrete XML schemas, this specification uses a convention where each
129 member of an element's [children] or [attributes] property is described using an XPath-like
130 notation (e.g., /x:MyHeader/x:SomeProperty/@value1). The use of {any} indicates the presence
131 of an element wildcard (<xs:any/>). The use of @{any} indicates the presence of an attribute
132 wildcard (<xs:anyAttribute/>).

133

134 2.2 Namespaces

135 Namespace URIs (of the general form "some-URI") represents some application-dependent or
136 context-dependent URI as defined in RFC 3986 [URI]. This specification is designed to work with
137 the general SOAP [SOAP11, SOAP12] message structure and message processing model, and
138 should be applicable to any version of SOAP. The current SOAP 1.1 namespace URI is used
139 herein to provide detailed examples, but there is no intention to limit the applicability of this
140 specification to a single version of SOAP.

141

142 The namespaces used in this document are shown in the following table (note that for brevity, the
143 examples use the prefixes listed below but do not include the URIs – those listed below are
144 assumed).

145

146 `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-`
147 `1.0.xsd`

148 `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-`
149 `1.0.xsd`

150 `http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd`

151 The following namespace prefixes are used in this document:

Prefix	Namespace
S11	<code>http://schemas.xmlsoap.org/soap/envelope/</code>

S12	http://www.w3.org/2003/05/soap-envelope
ds	http://www.w3.org/2000/09/xmlsig#
xenc	http://www.w3.org/2001/04/xmlenc#
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
wsse11	http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd

152

Table 1- Namespace prefixes

153 URI fragments defined in this specification are relative to the following base URI unless otherwise
154 stated:

155

156 [http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0)
157 [profile-1.0](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0)

158

159 The following table lists the full URI for each URI fragment referred to in this specification.

URI Fragment	Full URI
#Base64Binary	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary
#STR-Transform	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#STR-Transform
#PKCS7	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#PKCS7
#X509v3	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3
#X509SubjectKeyIdentifier	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509SubjectKeyIdentifier

160

161 2.3 Terminology

162 This specification adopts the terminology defined in Web Services Security: SOAP Message
163 Security specification [WS-Security].

164

165 Readers are presumed to be familiar with the definitions of terms in the Internet Security Glossary
166 [Glossary].

167 3 Usage (Normative)

168 This specification describes the syntax and processing rules for the use of the X.509
169 authentication framework with the Web Services Security: SOAP Message Security specification
170 [WS-Security]. For the purposes of determining the order of preference of reference types, the
171 use of IssuerSerial within X509Data should be considered to be a form of Key Identifier

172 3.1 Token types

173 This profile defines the syntax of, and processing rules for, three types of binary security token
174 using the URI values specified in Table 2.

175

176 If the `ValueType` attribute is missing, the receiver may interpret it either based on a prior
177 agreement or by parsing the content.

178

Token	ValueType URI	Description
Single certificate	#X509v3	An X.509 v3 certificate capable of signature-verification at a minimum
Certificate Path	#X509PKIPathv1	An ordered list of X.509 certificates packaged in a PKIPath
Set of certificates and CRLs	#PKCS7	A list of X.509 certificates and (optionally) CRLs packaged in a PKCS#7 wrapper

179

Table 2 – Token types

180 3.1.1 X509v3 Token Type

181 The type of the end-entity that is authenticated by a certificate used in this manner is a matter of
182 policy that is outside the scope of this specification.

183 3.1.2 X509PKIPathv1 Token Type

184 The `X509PKIPathv1` token type MAY be used to represent a certificate path.

185 3.1.3 PKCS7 Token Type

186 The `PKCS7` token type MAY be used to represent a certificate path. It is RECOMMENDED that
187 applications use the `PKIPath` object for this purpose instead.

188

189 The order of the certificates in a `PKCS#7` data structure is not significant. If an ordered certificate
190 path is converted to `PKCS#7` encoded bytes and then converted back, the order of the

191 certificates may not be preserved. Processors SHALL NOT assume any significance to the order
192 of the certificates in the data structure. See [PKCS7] for more information.

193 3.2 Token References

194 In order to ensure a consistent processing model across all the token types supported by WSS:
195 SOAP Message Security, the `<wsse:SecurityTokenReference>` element SHALL be used to
196 specify all references to X.509 token types in signature or encryption elements that comply with
197 this profile.

198

199 A `<wsse:SecurityTokenReference>` element MAY reference an X.509 token type by one of
200 the following means:

201

- 202 • Reference to a Subject Key Identifier

203 The `<wsse:SecurityTokenReference>` element contains a
204 `<wsse:KeyIdentifier>` element that specifies the token data by means of a
205 X.509 SubjectKeyIdentifier reference. A subject key identifier MAY only be used to
206 reference an X.509v3 certificate.”

207

- 208 • Reference to a Binary Security Token

209 The `<wsse:SecurityTokenReference>` element contains a `wsse:Reference>`
210 element that references a local `<wsse:BinarySecurityToken>` element or a
211 remote data source that contains the token data itself.

212

- 213 • Reference to an Issuer and Serial Number

214 The `<wsse:SecurityTokenReference>` element contains a `<ds:X509Data>`
215 element that contains a `<ds:X509IssuerSerial>` element that uniquely identifies
216 an end entity certificate by its X.509 Issuer and Serial Number.

217 3.2.1 Reference to an X.509 Subject Key Identifier

218 The `<wsse:KeyIdentifier>` element is used to specify a reference to an X.509v3 certificate
219 by means of a reference to its X.509 SubjectKeyIdentifier attribute. This profile defines the syntax
220 of, and processing rules for referencing a Subject Key Identifier using the URI values specified in
221 Table 3 (note that URI fragments are relative to `http://docs.oasis-`
222 `open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0`).

223

Subject Key Identifier	ValueType URI	Description
Certificate Key Identifier	<code>#X509SubjectKeyIdentifier</code>	Value of the certificate's X.509 SubjectKeyIdentifier

224

Table 3 – Subject Key Identifier

225 The `<wsse:SecurityTokenReference>` element from which the reference is made contains
226 the `<wsse:KeyIdentifier>` element. The `<wsse:KeyIdentifier>` element MUST have a
227 `ValueType` attribute with the value `#X509SubjectKeyIdentifier` and its contents MUST be

228 the value of the certificate's X.509v3 SubjectKeyIdentifier extension, encoded as per the
229 <wsse:KeyIdentifier> element's EncodingType attribute. For the purposes of this
230 specification, the value of the SubjectKeyIdentifier extension is the contents of the KeyIdentifier
231 octet string, excluding the encoding of the octet string prefix.

232 **3.2.2 Reference to a Security Token**

233 The <wsse:Reference> element is used to reference an X.509 security token value by means of
234 a URI reference.

235

236 The URI reference MAY be internal in which case the URI reference SHOULD be a bare name
237 XPointer reference to a <wsse:BinarySecurityToken> element contained in a preceding
238 message header that contains the binary X.509 security token data.

239 **3.2.3 Reference to an Issuer and Serial Number**

240 The <ds:X509IssuerSerial> element is used to specify a reference to an X.509 security
241 token by means of the certificate issuer name and serial number.

242

243 The <ds:X509IssuerSerial> element is a direct child of the <ds:X509Data> element that is
244 in turn a direct child of the <wsse:SecurityTokenReference> element in which the
245 reference is made

246 **3.3 Signature**

247 Signed data MAY specify the certificate associated with the signature using any of the X.509
248 security token types and references defined in this specification.

249

250 An X.509 certificate specifies a binding between a public key and a set of attributes that includes
251 (at least) a subject name, issuer name, serial number and validity interval. Other attributes may
252 specify constraints on the use of the certificate or affect the recourse that may be open to a
253 relying party that depends on the certificate. A given public key may be specified in more than
254 one X.509 certificate; consequently a given public key may be bound to two or more distinct sets
255 of attributes.

256

257 It is therefore necessary to ensure that a signature created under an X.509 certificate token
258 uniquely and irrefutably specifies the certificate under which the signature was created.

259

260 Implementations SHOULD protect against a certificate substitution attack by including either the
261 certificate itself or an immutable and unambiguous reference to the certificate within the scope of
262 the signature according to the method used to reference the certificate as described in the
263 following sections.

264 **3.3.1 Key Identifier**

265 The <wsse:KeyIdentifier> element does not guarantee an immutable and unambiguous
266 reference to the certificate referenced. Consequently implementations that use this form of
267 reference within a signature SHOULD employ the STR Dereferencing Transform within a

268 reference to the signature key information in order to ensure that the referenced certificate is
269 signed, and not just the ambiguous reference. The form of the reference is a bare name
270 reference as defined by the XPointer specification [XPointer].

271

272 The following example shows a certificate referenced by means of a KeyIdentifier. The scope of
273 the signature is the <ds:SignedInfo> element which includes both the message body (#body)
274 and the signing certificate by means of a reference to the <ds:KeyInfo> element which
275 references it (#keyinfo). Since the <ds:KeyInfo> element only contains a mutable reference to
276 the certificate rather than the certificate itself, a transformation is specified which replaces the
277 reference to the certificate with the certificate. The <ds:KeyInfo> element specifies the signing
278 key by means of a <wsse:SecurityTokenReference> element which contains a
279 <wsse:KeyIdentifier> element which specifies the X.509 subject key identifier of the signing
280 certificate.

281

```
282 <S11:Envelope xmlns:S11="...">
283   <S11:Header>
284     <wsse:Security
285       xmlns:wsse="..."
286       xmlns:wsu="...">
287       <ds:Signature
288         xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
289         <ds:SignedInfo>...
290         <ds:Reference URI="#body">...</ds:Reference>
291         <ds:Reference URI="#keyinfo">
292           <ds:Transforms>
293             <ds:Transform Algorithm="...#STR-Transform">
294               <wsse:TransformationParameters
295                 <ds:CanonicalizationMethod Algorithm="..."/>
296               </wsse:TransformationParameters>
297             </ds:Transform>
298           </ds:Transforms>...
299         </ds:Reference>
300       </ds:SignedInfo>
301       <ds:SignatureValue>HFLP...</ds:SignatureValue>
302       <ds:KeyInfo Id="keyinfo">
303         <wsse:SecurityTokenReference>
304           <wsse:KeyIdentifier EncodingType="...#Base64Binary"
305             ValueType="...#X509SubjectKeyIdentifier">
306             MIGfMa0GCSq...
307           </wsse:KeyIdentifier>
308         </wsse:SecurityTokenReference>
309       </ds:KeyInfo>
310     </ds:Signature>
311   </wsse:Security>
312 </S11:Header>
313 <S11:Body wsu:Id="body"
314   xmlns:wsu=".../">
315   ...
316 </S11:Body>
317 </S11:Envelope>
```

318 3.3.2 Reference to a Binary Security Token

319 The signed data SHOULD contain a core bare name reference (as defined by the XPointer
320 specification [XPointer]) to the <wsse:BinarySecurityToken> element that contains the
321 security token referenced, or a core reference to the external data source containing the security
322 token.

323

324 The following example shows a certificate embedded in a <wsse:BinarySecurityToken>
325 element and referenced by URI within a signature. The certificate is included in the
326 <wsse:Security> header as a <wsse:BinarySecurityToken> element with identifier
327 binarytoken. The scope of the signature defined by a <ds:Reference> element within the
328 <ds:SignedInfo> element includes the signing certificate which is referenced by means of the
329 URI bare name pointer #binarytoken. The <ds:KeyInfo> element specifies the signing key
330 by means of a <wsse:SecurityTokenReference> element which contains a
331 <wsse:Reference> element which references the certificate by means of the URI bare name
332 pointer #binarytoken.

333

```
334 <S11:Envelope xmlns:S11="...">
335   <S11:Header>
336     <wsse:Security
337       xmlns:wsse="..."
338       xmlns:wsu="...">
339       <wsse:BinarySecurityToken
340         wsu:Id="binarytoken"
341         ValueType="...#X509v3"
342         EncodingType="...#Base64Binary">
343         MIIIEZzCCA9CgAwIBAgIQEmtJZc0...
344       </wsse:BinarySecurityToken>
345       <ds:Signature
346         xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
347         <ds:SignedInfo>...
348         <ds:Reference URI="#body">...</ds:Reference>
349         <ds:Reference URI="#binarytoken">...</ds:Reference>
350       </ds:SignedInfo>
351       <ds:SignatureValue>HFLP...</ds:SignatureValue>
352       <ds:KeyInfo>
353         <wsse:SecurityTokenReference>
354           <wsse:Reference URI="#binarytoken" />
355         </wsse:SecurityTokenReference>
356       </ds:KeyInfo>
357     </ds:Signature>
358   </wsse:Security>
359 </S11:Header>
360 <S11:Body wsu:Id="body"
361   xmlns:wsu="...">
362   ...
363 </S11:Body>
364 </S11:Envelope>
```

365 3.3.3 Reference to an Issuer and Serial Number

366 The signed data SHOULD contain a core bare name reference (as defined by the XPointer
367 specification [XPointer]) to the <ds:KeyInfo> element that contains the security token
368 reference.

369

370 The following example shows a certificate referenced by means of its issuer name and serial
371 number. In this example the certificate is not included in the message. The scope of the signature
372 defined by the <ds:SignedInfo> element includes both the message body (#body) and the key
373 information element (#keyInfo). The <ds:KeyInfo> element contains a
374 <wsse:SecurityTokenReference> element which specifies the issuer and serial number of
375 the specified certificate by means of the <ds:X509IssuerSerial> element.

376

```
377 <S11:Envelope xmlns:S11="...">
378   <S11:Header>
379     <wsse:Security
380       xmlns:wsse="..."
381       xmlns:wsmu="...">
382       <ds:Signature
383         xmlns:ds="...">
384         <ds:SignedInfo>...
385           <ds:Reference URI="#body">...</ds:Reference>
386           <ds:Reference URI="#keyinfo">...</ds:Reference>
387         </ds:SignedInfo>
388         <ds:SignatureValue>HFLP...</ds:SignatureValue>
389         <ds:KeyInfo Id="keyinfo">
390           <wsse:SecurityTokenReference>
391             <ds:X509Data>
392               <ds:X509IssuerSerial>
393                 <ds:X509IssuerName>
394                   DC=ACMECorp, DC=com
395                 </ds:X509IssuerName>
396                 <ds:X509SerialNumber>12345678</ds:X509SerialNumber>
397               </ds:X509IssuerSerial>
398             </ds:X509Data>
399           </wsse:SecurityTokenReference>
400         </ds:KeyInfo>
401       </ds:Signature>
402     </wsse:Security>
403   </S11:Header>
404   <S11:Body wsu:Id="body"
405     xmlns:wsmu="...">
406     ...
407   </S11:Body>
408 </S11:Envelope>
```

409 3.4 Encryption

410 Encrypted keys or data MAY identify a key required for decryption by identifying the
411 corresponding key used for encryption by means of any of the X.509 security token types or
412 references specified herein.

413

414 Since the sole purpose is to identify the decryption key it is not necessary to specify either a trust
415 path or the specific contents of the certificate itself.

416

417 The following example shows a decryption key referenced by means of the issuer name and
418 serial number of an associated certificate. In this example the certificate is not included in the
419 message. The <ds:KeyInfo> element contains a <wsse:SecurityTokenReference>
420 element which specifies the issuer and serial number of the specified certificate by means of the
421 <ds:X509IssuerSerial> element.

422

```
423 <S11:Envelope  
424     xmlns:S11="..."  
425     xmlns:ds="..."  
426     xmlns:wsse="..."  
427     xmlns:xenc="...">  
428   <S11:Header>  
429     <wsse:Security>  
430       <xenc:EncryptedKey>  
431         <xenc:EncryptionMethod Algorithm="..."/>  
432         <ds:KeyInfo>  
433           <wsse:SecurityTokenReference>  
434             <ds:X509Data>  
435               <ds:X509IssuerSerial>  
436                 <ds:X509IssuerName>  
437                   DC=ACMECorp, DC=com  
438                 </ds:X509IssuerName>  
439                 <ds:X509SerialNumber>12345678</ds:X509SerialNumber>  
440               </ds:X509IssuerSerial>  
441             </ds:X509Data>  
442           </wsse:SecurityTokenReference>  
443         </ds:KeyInfo>  
444         <xenc:CipherData>  
445           <xenc:CipherValue>...</xenc:CipherValue>  
446         </xenc:CipherData>  
447         <xenc:ReferenceList>  
448           <xenc:DataReference URI="#encrypted"/>  
449         </xenc:ReferenceList>  
450       </xenc:EncryptedKey>  
451     </wsse:Security>  
452   </S11:Header>  
453   <S11:Body>  
454     <xenc:EncryptedData Id="encrypted" Type="...">  
455       <xenc:CipherData>  
456         <xenc:CipherValue>...</xenc:CipherValue>  
457       </xenc:CipherData>  
458     </xenc:EncryptedData>  
459   </S11:Body>  
460 </S11:Envelope>
```

461

462 The following example shows a decryption key referenced by means of the Thumbprint of an
463 associated certificate. In this example the certificate is not included in the message. The
464 <ds:KeyInfo> element contains a <wsse:SecurityTokenReference> element which
465 specifies the Thumbprint of the specified certificate by means of the <http://docs.oasis->

466 open.org/wss/oasis-wss-soap-message-security-1.1#ThumbprintSHA1 attribute of
467 the <wsse:KeyIdentifier> element.

```
468 <S11:Envelope
469   xmlns:S11="..."
470   xmlns:ds="..."
471   xmlns:wsse="..."
472   xmlns:xenc="...">
473   <S11:Header>
474     <wsse:Security>
475       <xenc:EncryptedKey>
476         <xenc:EncryptionMethod Algorithm="..." />
477         <ds:KeyInfo>
478           <wsse:SecurityTokenReference>
479             <wsse:KeyIdentifier
480               ValueType="http://docs.oasis-open.org/wss/oasis-wss-
481 soap-message-security-1.1#ThumbprintSHA1" >LKiQ/CmFrJDJqCLFcjIhIsmZ/+0=
482             </wsse:KeyIdentifier>
483           </wsse:SecurityTokenReference>
484         </ds:KeyInfo>
485       <xenc:CipherData>
486         <xenc:CipherValue>...</xenc:CipherValue>
487       </xenc:CipherData>
488       <xenc:ReferenceList>
489         <xenc:DataReference URI="#encrypted" />
490       </xenc:ReferenceList>
491     </xenc:EncryptedKey>
492   </wsse:Security>
493 </S11:Header>
494 <S11:Body>
495   <xenc:EncryptedData Id="encrypted" Type="...">
496     <xenc:CipherData>
497       <xenc:CipherValue>...</xenc:CipherValue>
498     </xenc:CipherData>
499   </xenc:EncryptedData>
500 </S11:Body>
501 </S11:Envelope>
```

502

503 3.5 Error Codes

504 When using X.509 certificates, the error codes defined in the WSS: SOAP Message Security
505 specification [WS-Security] MUST be used.

506

507 If an implementation requires the use of a custom error it is recommended that a sub-code be
508 defined as an extension of one of the codes defined in the WSS: SOAP Message Security
509 specification [WS-Security].

510

511 **4 Threat Model and Countermeasures (Non-**
512 **Normative)**

513 The use of X.509 certificate token introduces no new threats beyond those identified in WSS:
514 SOAP Message Security specification [WS-Security].

515

516 Message alteration and eavesdropping can be addressed by using the integrity and confidentiality
517 mechanisms described in WSS: SOAP Message Security [WS-Security]. Replay attacks can be
518 addressed by using message timestamps and caching, as well as other application-specific
519 tracking mechanisms. For X.509 certificates, identity is authenticated by use of keys, man-in-the-
520 middle attacks are generally mitigated.

521

522 It is strongly RECOMMENDED that all relevant and immutable message data be signed.

523

524 It should be noted that a transport-level security protocol such as SSL or TLS [RFC2246] MAY be
525 used to protect the message and the security token as an alternative to or in conjunction with
526 WSS: SOAP Message Security specification [WS-Security].

527

5 References

528 The following are normative references

- 529 **[Glossary]** Informational RFC 2828, *Internet Security Glossary*, May 2000.
530 <http://www.ietf.org/rfc/rfc2828.txt>
- 531 **[KEYWORDS]** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
532 RFC 2119, Harvard University, March 1997,
533 <http://www.ietf.org/rfc/rfc2119.txt>
- 534 **[RFC2246]** T. Dierks, C. Allen., *The TLS Protocol Version, 1.0*. IETF RFC 2246
535 January 1999. <http://www.ietf.org/rfc/rfc2246.txt>
- 536 **[SOAP11]** W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May 2000.
- 537 **[SOAP12]** W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging
538 Framework", 23 June 2003.
- 539 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers
540 (URI): Generic Syntax," RFC 3986, MIT/LCS, Day Software, Adobe
541 Systems, January 2005.
- 542 **[WS-Security]** A. Nadalin et al., *Web Services Security: SOAP Message Security 1.1*
543 (WS-Security 2004), OASIS Standard, [http://docs.oasis-](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.1.pdf)
544 [open.org/wss/2004/01/oasis-200401-wss-soap-message-security-](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.1.pdf)
545 [1.1.pdf](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.1.pdf).
- 546 **[PKCS7]** *PKCS #7: Cryptographic Message Syntax Standard* RSA Laboratories,
547 November 1, 1993. [http://www.rsasecurity.com/rsalabs/pkcs/pkcs-](http://www.rsasecurity.com/rsalabs/pkcs/pkcs-7/index.html)
548 [7/index.html](http://www.rsasecurity.com/rsalabs/pkcs/pkcs-7/index.html)
- 549 **[PKIPATH]** [http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-](http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-REC-X.509-200110-S!Cor1)
550 [REC-X.509-200110-S!Cor1](http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-REC-X.509-200110-S!Cor1)
- 551 **[X509]** ITU-T Recommendation X.509 (1997 E): *Information Technology - Open*
552 *Systems Interconnection - The Directory: Authentication Framework*,
553 June 1997.

554

555 The following are non-normative references

- 556 **[XML-ns]** T. Bray, D. Hollander, A. Layman. *Namespaces in XML. W3C*
557 *Recommendation*. January 1999. [http://www.w3.org/TR/1999/REC-xml-](http://www.w3.org/TR/1999/REC-xml-names-19990114)
558 [names-19990114](http://www.w3.org/TR/1999/REC-xml-names-19990114)
- 559 **[XML Encrypt]** W3C Recommendation, "XML Encryption Syntax and Processing," 10
560 December 2002
- 561 **[XML Signature]** D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. *XML-*
562 *Signature Syntax and Processing*, W3C Recommendation, 12 February
563 2002.

564

565

Appendix A: Acknowledgments

Current Contributors:

Michael	Hu	Actional
Maneesh	Sahu	Actional
Duane	Nickull	Adobe Systems
Gene	Thurston	AmberPoint
Frank	Siebenlist	Argonne National Laboratory
Hal	Lockhart	BEA Systems
Denis	Pilipchuk	BEA Systems
Corinna	Witt	BEA Systems
Steve	Anderson	BMC Software
Rich	Levinson	Computer Associates
Thomas	DeMartini	ContentGuard
Merlin	Hughes	Cybertrust
Dale	Moberg	Cyclone Commerce
Rich	Salz	Datapower
Sam	Wei	EMC
Dana S.	Kaufman	Forum Systems
Toshihiro	Nishimura	Fujitsu
Kefeng	Chen	GeoTrust
Irving	Reid	Hewlett-Packard
Kojiro	Nakayama	Hitachi
Paula	Austel	IBM
Derek	Fu	IBM
Maryann	Hondo	IBM
Kelvin	Lawrence	IBM
Michael	McIntosh	IBM
Anthony	Nadalin	IBM
Nataraj	Nagaratnam	IBM
Bruce	Rich	IBM
Ron	Williams	IBM
Don	Flinn	Individual
Kate	Cherry	Lockheed Martin
Paul	Cotton	Microsoft
Vijay	Gajjala	Microsoft
Martin	Gudgin	Microsoft
Chris	Kaler	Microsoft
Frederick	Hirsch	Nokia
Abbie	Barbir	Nortel
Prateek	Mishra	Oracle
Vamsi	Motukuru	Oracle
Ramana	Turlapi	Oracle
Ben	Hammond	RSA Security
Rob	Philpott	RSA Security
Blake	Dournaee	Sarvega
Sundeep	Pechu	Sarvega

Coumara	Radja	Sarvega
Pete	Wenzel	SeeBeyond
Manveen	Kaur	Sun Microsystems
Ronald	Monzillo	Sun Microsystems
Jan	Alexander	Systinet
Symon	Chang	TIBCO Software
John	Weiland	US Navy
Hans	Granqvist	VeriSign
Phillip	Hallam-Baker	VeriSign
Hemma	Prafullchandra	VeriSign

569

Previous Contributors:

Peter	Dapkus	BEA
Guillermo	Lao	ContentGuard
TJ	Pannu	ContentGuard
Xin	Wang	ContentGuard
Shawn	Sharp	Cyclone Commerce
Ganesh	Vaideeswaran	Documentum
Tim	Moses	Entrust
Carolina	Canales-Valenzuela	Ericsson
Tom	Rutt	Fujitsu
Yutaka	Kudo	Hitachi
Jason	Rouault	HP
Bob	Blakley	IBM
Joel	Farrell	IBM
Satoshi	Hada	IBM
Hiroshi	Maruyama	IBM
David	Melgar	IBM
Kent	Tamura	IBM
Wayne	Vicknair	IBM
Phil	Griffin	Individual
Mark	Hayes	Individual
John	Hughes	Individual
Peter	Rostin	Individual
Davanum	Srinivas	Individual
Bob	Morgan	Individual/Internet2
Bob	Atkinson	Microsoft
Keith	Ballinger	Microsoft
Allen	Brown	Microsoft
Giovanni	Della-Libera	Microsoft
Alan	Geller	Microsoft
Johannes	Klein	Microsoft
Scott	Konersmann	Microsoft
Chris	Kurt	Microsoft
Brian	LaMacchia	Microsoft
Paul	Leach	Microsoft
John	Manferdelli	Microsoft
John	Shewchuk	Microsoft
Dan	Simon	Microsoft

Hervey	Wilson	Microsoft
Jeff	Hodges	Neustar
Senthil	Sengodan	Nokia
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Vipin	Samar	Oracle
Jerry	Schwarz	Oracle
Eric	Gravengaard	Reactivity
Andrew	Nash	Reactivity
Stuart	King	Reed Elsevier
Martijn	de Boer	SAP
Jonathan	Tourzan	Sony
Yassir	Elley	Sun
Michael	Nguyen	The IDA of Singapore
Don	Adams	TIBCO
Morten	Jorgensen	Vordel

570

571

Appendix B: Revision History

Rev	Date	By Whom	What
errata	08-25-2006	Anthony Nadalin	Issue 457, 458, 460

572