



Web Services Security X.509 Certificate Token Profile 1.1

Committee Draft – Tuesday 14, June 2005

OASIS Identifier:

{WSS: SOAP Message Security }-{X509 Profile }-{1.0} (Word) (PDF)

Document Location:

<http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-wss-x509-token-profile-1.1>

Errata Location:

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss

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 interim draft.

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

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 WS-Security TC web page (<http://www.oasis-open.org/committees/wss/ipr.php>).

35 **Notices**

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

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

48 Copyright © The Organization for the Advancement of Structured Information Standards [OASIS]
49 2002-2005. All Rights Reserved.

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

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

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

66 **Table of Contents**

67 1 Introduction (Non-Normative) 4
68 2 Notations and Terminology (Normative) 5
69 2.1 Notational Conventions 5
70 2.2 Namespaces 5
71 2.3 Terminology 6
72 3 Usage (Normative) 8
73 3.1 Token types 8
74 3.1.1 X509 Token Type 8
75 3.1.2 X509PKIPathv1 Token Type 8
76 3.1.3 PKCS7 Token Type 8
77 3.2 Token References 9
78 3.2.1 Reference to an X.509 Subject Key Identifier 9
79 3.2.2 Reference to a Security Token 10
80 3.2.3 Reference to an Issuer and Serial Number 10
81 3.2.4 Thumbprint References 10
82 3.3 Signature 10
83 3.3.1 Key Identifier 11
84 3.3.2 Reference to a Binary Security Token 12
85 3.3.3 Reference to an Issuer and Serial Number 13
86 3.4 Encryption 14
87 3.5 Error Codes 15
88 4 Threat Model and Countermeasures (Non-Normative) 16
89 5 References 17
90 Appendix A: Acknowledgments **Error! Bookmark not defined.**
91 Appendix B: Revision History 20
92

93 **1 Introduction (Non-Normative)**

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

96

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

101

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

104

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

2 Notations and Terminology (Normative)

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

2.1 Notational Conventions

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

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

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

2.2 Namespaces

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

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

```
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd
http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-wss-wssecurity-secext-1.1.xsd
```

The following namespace prefixes are used in this document:

Prefix	Namespace
--------	-----------

S11	http://schemas.xmlsoap.org/soap/envelope/
S12	http://www.w3.org/2003/05/soap-envelope
ds	http://www.w3.org/2000/09/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/2005/xx/oasis-2005xx-wss-wssecurity-secext-1.1.xsd
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd

144

Table 1- Namespace prefixes

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

147

148 [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)
149 [profile-1.0](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0)

150

151 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
#X509PKIPathv1	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509PKIPathv1
#X509SubjectKeyIdentifier	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509SubjectKeyIdentifier

152

153 2.3 Terminology

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

156

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

159 3 Usage (Normative)

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

164 3.1 Token types

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

167

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

170

Token	ValueType URI	Description
Single certificate	#X509v3	An X.509 v3 signature-verification certificate
Single certificate	#x509v1	An X.509 v1 signature-verification certificate.
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

171

Table 2 – Token types

172 3.1.1 X509v3 Token Type

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

175 3.1.2 X509PKIPathv1 Token Type

176 The `x509PKIPathv1` token type MAY be used to represent a certificate path.

177 3.1.3 PKCS7 Token Type

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

180

181 The order of the certificates in a PKCS#7 data structure is not significant. If an ordered certificate
 182 path is converted to PKCS#7 encoded bytes and then converted back, the order of the
 183 certificates may not be preserved. Processors SHALL NOT assume any significance to the order
 184 of the certificates in the data structure. See [PKCS7] for more information.

185 3.2 Token References

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

190

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

193

194 Reference to a Subject Key Identifier

195 The <wsse:SecurityTokenReference> element contains a
 196 <wsse:KeyIdentifier> element that specifies the token data by means of a X.509
 197 SubjectKeyIdentifier reference. A subject key identifier may only be used to reference an
 198 X.509v3 certificate.”

199

200 Reference to a Binary Security Token

201 The <wsse:SecurityTokenReference> element contains a <wsse:Reference>
 202 element that references a local <wsse:BinarySecurityToken> element or a remote
 203 data source that contains the token data itself.

204

205 Reference to an Issuer and Serial Number

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

209 3.2.1 Reference to an X.509 Subject Key Identifier

210 The <wsse:KeyIdentifier> element is used to specify a reference to an X.509v3 certificate
 211 by means of a reference to its X.509 SubjectKeyIdentifier attribute. This profile defines the syntax
 212 of, and processing rules for referencing a Subject Key Identifier using the URI values specified in
 213 Table 3 (note that URI fragments are relative to the URI for this specification).

214

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

215

Table 3 – Subject Key Identifier

216 The <wsse:SecurityTokenReference> element from which the reference is made contains
 217 the <wsse:KeyIdentifier> element. The <wsse:KeyIdentifier> element MUST have a
 218 ValueType attribute with the value #X509SubjectKeyIdentifier and its contents MUST be the
 219 value of the certificate's X.509v3 SubjectKeyIdentifier extension, encoded as per the

220 <wsse:KeyIdentifier> element's EncodingType attribute. For the purposes of this
221 specification, the value of the SubjectKeyIdentifier extension is the contents of the KeyIdentifier
222 octet string, excluding the encoding of the octet string prefix.

223 3.2.2 Reference to a Security Token

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

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

229 3.2.3 Reference to an Issuer and Serial Number

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

232

233 The <ds:X509IssuerSerial> element is a direct child of the <ds:X509Data> element that is
234 in turn a direct child of the <wsse:SecurityTokenReference> element in which the
235 reference is made.

236 3.2.4 Thumbprint References

237 The <wsse:KeyIdentifier> element is used to specify a reference to an X.509 certificate by
238 means of a reference to its X.509 Thumbprint attribute. This profile defines the syntax of, and the
239 processing rules for referencing a Thumbprint using the URI values specified below (note that the
240 URI fragment is relative to [http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-
241 wss-soap-message-security-1.1](http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-wss-soap-message-security-1.1)):

242

Subject Key Identifier	ValueType URI	Description
Thumbprint	#X509ThumbprintSHA1	The thumbprint of the X.509 certificate

243

244 The <wsse:SecurityTokenReference> element from which the reference is made contains a
245 <wsse:KeyIdentifier> element. The <wsse:KeyIdentifier> element MUST have a
246 ValueType attribute with the value or [http://docs.oasis-
247 open.org/wss/2005/xx/oasis-2005xx-wss-soap-message-security-
248 1.1#ThumbprintSHA1](http://docs.oasis-open.org/wss/2005/xx/oasis-2005xx-wss-soap-message-security-1.1#ThumbprintSHA1) and its contents MUST be the thumbprint for the desired certificate . If
249 the certificate does not contain a X.509 Thumbprint extension, then one is computed as the SHA1
250 of the raw octets which would be encoded within the <wsse:BinarySecurityToken> element
251 were it to be included. The thumbprint is encoded as per the <wsse:KeyIdentifier>
252 element's EncodingType attribute. The default encoding is Base64. Implementations compliant
253 with this specification MAY support such a certificate reference mechanism.

254 3.3 Signature

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

257

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

264

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

267

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

272 3.3.1 Key Identifier

273 The <wsse:KeyIdentifier> element does not guarantee an immutable and unambiguous
274 reference to the certificate referenced. Consequently implementations that use this form of
275 reference within a signature SHOULD employ the STR Dereferencing Transform within a
276 reference to the signature key information in order to ensure that the referenced certificate is
277 signed, and not just the ambiguous reference. The form of the reference is a bare name
278 reference as defined by the XPointer specification [XPointer].

279

280 The following example shows a certificate referenced by means of a KeyIdentifier. The scope of
281 the signature is the <ds:SignedInfo> element which includes both the message body (#body)
282 and the signing certificate by means of a reference to the <ds:KeyInfo> element which
283 references it (#keyinfo). Since the <ds:KeyInfo> element only contains a mutable reference to
284 the certificate rather than the certificate itself, a transformation is specified which replaces the
285 reference to the certificate with the certificate. The <ds:KeyInfo> element specifies the signing
286 key by means of a <wsse:SecurityTokenReference> element which contains a
287 <wsse:KeyIdentifier> element which specifies the X.509 subject key identifier of the signing
288 certificate.

289

```
290 <S11:Envelope xmlns:S11="...">
291   <S11:Header>
292     <wsse:Security
293       xmlns:wsse="..."
294       xmlns:wssu="...">
295       <ds:Signature
296         xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
297         <ds:SignedInfo>...
298         <ds:Reference URI="#body">...</ds:Reference>
299         <ds:Reference URI="#keyinfo">
300           <ds:Transforms>
301             <ds:Transform Algorithm="...#STR-Transform">
302               <wsse:TransformationParameters>
303                 <ds:CanonicalizationMethod Algorithm="..." />
304               </wsse:TransformationParameters>
```

```

305         </ds:Transform>
306     </ds:Transforms>...
307 </ds:Reference>
308 </ds:SignedInfo>
309 <ds:SignatureValue>HFLP...</ds:SignatureValue>
310 <ds:KeyInfo Id="keyinfo">
311     <wsse:SecurityTokenReference>
312         <wsse:KeyIdentifier EncodingType="...#Base64Binary"
313             ValueType="...#X509SubjectKeyIdentifier">
314             MIGfMa0GCSq...
315         </wsse:KeyIdentifier>
316     </wsse:SecurityTokenReference>
317 </ds:KeyInfo>
318 </ds:Signature>
319 </wsse:Security>
320 </S11:Header>
321 <S11:Body wsu:Id="body"
322     xmlns:wsu=".../">
323     ...
324 </S11:Body>
325 </S11:Envelope>

```

326 3.3.2 Reference to a Binary Security Token

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

331

332 The following example shows a certificate embedded in a <wsse:BinarySecurityToken>
333 element and referenced by URI within a signature. The certificate is included in the
334 <wsse:Security> header as a <wsse:BinarySecurityToken> element with identifier
335 binarytoken. The scope of the signature defined by a <ds:Reference> element within the
336 <ds:SignedInfo> element includes the signing certificate which is referenced by means of the
337 URI bare name pointer #binarytoken. The <ds:KeyInfo> element specifies the signing key
338 by means of a <wsse:SecurityTokenReference> element which contains a
339 <wsse:Reference> element which references the certificate by means of the URI bare name
340 pointer #binarytoken.

```

341 <S11:Envelope xmlns:S11="...">
342     <S11:Header>
343         <wsse:Security
344             xmlns:wsse="..."
345             xmlns:wsu="...">
346             <wsse:BinarySecurityToken
347                 wsu:Id="binarytoken"
348                 ValueType="...#X509v3"
349                 EncodingType="...#Base64Binary">
350                 MIEEZzCCA9CgAwIBAgIQEmtJZc0...
351             </wsse:BinarySecurityToken>
352             <ds:Signature
353                 xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
354                 <ds:SignedInfo>...
355                 <ds:Reference URI="#body">...</ds:Reference>
356                 <ds:Reference URI="#binarytoken">...</ds:Reference>

```

```

357     </ds:SignedInfo>
358     <ds:SignatureValue>HFLP...</ds:SignatureValue>
359     <ds:KeyInfo>
360         <wsse:SecurityTokenReference>
361             <wsse:Reference URI="#binarytoken" />
362         </wsse:SecurityTokenReference>
363     </ds:KeyInfo>
364 </ds:Signature>
365 </wsse:Security>
366 </S11:Header>
367 <S11:Body wsu:Id="body"
368     xmlns:wsu="...">
369     ...
370 </S11:Body>
371 </S11:Envelope>

```

372 3.3.3 Reference to an Issuer and Serial Number

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

376

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

383

```

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

```

```
409     </wsse:Security>
410 </S11:Header>
411 <S11:Body wsu:Id="body"
412     xmlns:wsu="...">
413     ...
414 </S11:Body>
415 </S11:Envelope>
```

3.4 Encryption

416 Encrypted keys or data MAY identify a key required for decryption by identifying the
417 corresponding key used for encryption by means of any of the X.509 security token types or
418 references specified herein.
419

420

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

423

424 It is RECOMMENDED that implementations specify an encryption key by reference to the Issuer
425 and Serial Number of an X509v3 certificate security token.

426

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

432

```
433 <S11:Envelope
434     xmlns:S11="..."
435     xmlns:ds="..."
436     xmlns:wsse="..."
437     xmlns:xenc="...">
438 <S11:Header>
439 <wsse:Security>
440 <xenc:EncryptedKey>
441 <xenc:EncryptionMethod Algorithm="..." />
442 <ds:KeyInfo>
443 <wsse:SecurityTokenReference>
444 <ds:X509Data>
445 <ds:X509IssuerSerial>
446 <ds:X509IssuerName>
447     DC=ACMECorp, DC=com
448 </ds:X509IssuerName>
449 <ds:X509SerialNumber>12345678</X509SerialNumber>
450 </ds:X509IssuerSerial>
451 </ds:X509Data>
452 </wsse:SecurityTokenReference>
453 </ds:KeyInfo>
454 <xenc:CipherData>
455 <xenc:CipherValue>...</xenc:CipherValue>
456 </xenc:CipherData>
457 <xenc:ReferenceList>
458 <xenc:DataReference URI="#encrypted" />
```

```
459         </xenc:ReferenceList>
460     </xenc:EncryptedKey>
461 </wsse:Security>
462 </S11:Header>
463 <S11:Body>
464     <xenc:EncryptedData Id="encrypted" Type="...">
465         <xenc:CipherData>
466             <xenc:CipherValue>...</xenc:CipherValue>
467         </xenc:CipherData>
468     </xenc:EncryptedData>
469 </S11:Body>
470 </S11:Envelope>
```

471 **3.5 Error Codes**

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

474

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

478

479 **4 Threat Model and Countermeasures (Non-**
480 **Normative)**

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

483

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

489

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

491

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

5 References

495

496 The following are normative references

- 497 **[Glossary]** Informational RFC 2828, *Internet Security Glossary*, May 2000.
498 <http://www.ietf.org/rfc/rfc2828.txt>
- 499 **[KEYWORDS]** S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*,
500 RFC 2119, Harvard University, March 1997,
501 <http://www.ietf.org/rfc/rfc2119.txt>
- 502 **[RFC2246]** T. Dierks, C. Allen., *The TLS Protocol Version, 1.0*. IETF RFC 2246
503 January 1999. <http://www.ietf.org/rfc/rfc2246.txt>
- 504 **[SOAP11]** W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May 2000.
- 505 **[SOAP12]** W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging
506 Framework", 23 June 2003.
- 507 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers
508 (URI): Generic Syntax," RFC 3986, MIT/LCS, Day Software, Adobe
509 Systems, January 2005.

510

511 The following are non-normative references

- 512 **[WS-Security]** OASIS, "Web Services Security: SOAP Message Security" 19 January
513 2004, [http://www.docs.oasis-open.org/wss/2004/01/oasis-200401-wss-](http://www.docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0)
514 [soap-message-security-1.0](http://www.docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0)
- 515 **[XML-ns]** T. Bray, D. Hollander, A. Layman. *Namespaces in XML. W3C*
516 *Recommendation*. January 1999. [http://www.w3.org/TR/1999/REC-xml-](http://www.w3.org/TR/1999/REC-xml-names-19990114)
517 [names-19990114](http://www.w3.org/TR/1999/REC-xml-names-19990114)
- 518 **[XML Encrypt]** W3C Recommendation, "XML Encryption Syntax and Processing," 10
519 December 2002
- 520 **[XML Signature]** D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. *XML-*
521 *Signature Syntax and Processing*, W3C Recommendation, 12 February
522 2002. <http://www.w3.org/TR/xmlsig-core/>
- 523 **[PKCS7]** *PKCS #7: Cryptographic Message Syntax Standard* RSA Laboratories,
524 November 1, 1993. [http://www.rsasecurity.com/rsalabs/pkcs-](http://www.rsasecurity.com/rsalabs/pkcs/pkcs-7/index.html)
525 [7/index.html](http://www.rsasecurity.com/rsalabs/pkcs/pkcs-7/index.html)
- 526 **[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-!Cor1)
527 [REC-X.509-200110-!Cor1](http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-REC-X.509-200110-!Cor1)
- 528 **[X509]** ITU-T Recommendation X.509 (1997 E): *Information Technology - Open*
529 *Systems Interconnection - The Directory: Authentication Framework*,
530 June 1997.

531

532

Appendix A: Acknowledgments

Contributors:

Gene	Thurston	AmberPoint
Frank	Siebenlist	Argonne National Lab
Merlin	Hughes	Baltimore Technologies
Irving	Reid	Baltimore Technologies
Peter	Dapkus	BEA
Hal	Lockhart	BEA
Steve	Anderson	BMC (Sec)
Srinivas	Davanum	Computer Associates
Thomas	DeMartini	ContentGuard
Guillermo	Lao	ContentGuard
TJ	Pannu	ContentGuard
Shawn	Sharp	Cyclone Commerce
Ganesh	Vaideeswaran	Documentum
Sam	Wei	Documentum
John	Hughes	Entegrity
Tim	Moses	Entrust
Toshihiro	Nishimura	Fujitsu
Tom	Rutt	Fujitsu
Jason	Rouault	HP
Yutaka	Kudo	Hitachi
Paula	Austel	IBM
Maryann	Hondo	IBM
Michael	McIntosh	IBM
Kelvin	Lawrence	IBM (co-Chair)
Anthony	Nadalin	IBM
Nataraj	Nagaratnam	IBM
Don	Flinn	Individual
Bob	Morgan	Individual
Paul	Cotton	Microsoft
Vijay	Gajjala	Microsoft
Chris	Kaler	Microsoft (co-Chair)
Chris	Kurt	Microsoft
John	Shewchuk	Microsoft
Prateek	Mishra	Netegrity
Frederick	Hirsch	Nokia
Senthil	Sengodan	Nokia
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Vipin	Samar	Oracle
Jerry	Schwarz	Oracle
Eric	Gravengaard	Reactivity
Stuart	King	Reed Elsevier
Andrew	Nash	RSA Security
Rob	Philpott	RSA Security

Peter	Rostin	RSA Security
Martijn	de Boer	SAP
Blake	Dournaee	Sarvega
Pete	Wenzel	SeeBeyond
Jonathan	Tourzan	Sony
Yassir	Elley	Sun Microsystems
Jeff	Hodges	Sun Microsystems
Ronald	Monzillo	Sun Microsystems
Jan	Alexander	Systinet
Michael	Nguyen	The IDA of Singapore
Don	Adams	TIBCO
Symon	Chang	TIBCO
John	Weiland	US Navy
Phillip	Hallam-Baker	VeriSign
Morten	Jorgensen	Vordel

535

536 **Contributors of input documents (if not already listed above) :**

Bob	Blakley	IBM
Joel	Farrell	IBM
Satoshi	Hada	IBM
Hiroshi	Maruyama	IBM
David	Melgar	IBM
Bob	Atkinson	Microsoft
Allen	Brown	Microsoft
Giovanni	Della-Libera	Microsoft
Johannes	Klein	Microsoft
Scott	Konersmann	Microsoft
Brian	LaMacchia	Microsoft
Paul	Leach	Microsoft
John	Manferdelli	Microsoft
Dan	Simon	Microsoft
Hervey	Wilson	Microsoft
Hemma	Prafullchandra	VeriSign

537

Appendix B: Revision History

Rev	Date	By Whom	What
WGD 1.1	2004-09-13	Anthony Nadalin	Initial version cloned from the Vwesion 1.1 and Errata
WGD 1.1	2005-03-22	Anthony Nadalin	Issue 373
WGD 1.1	2005-05-11	Anthony Nadalin	Issue 388
WGD 1.1	2005-05-17	Anthony Nadalin	Formatting Issues
WGD 1.1	2005-06-14	Anthony Nadalin	Fix Example