



1

2 **Web Services Security**
3 **Rights Expression Language (REL)**
4 **Token Profile 1.1**

5 **Committee Specification: 14 November 2005**

6 **OASIS identifier:**

7 `wss-v1.1-spec-cs-REL-token-profile`

8 **Document Location:**

9 `http://docs.oasis-open.org/wss/oasis-wss-rel-token-profile-`
10 `1.1.pdf`

11 **Errata Location:**

12 `http://www.oasis-open.org/committees/wss`

13 **Technical Committee:**

14 `Web Services Security (WSS)`

15 **Chairs:**

16 `Kelvin Lawrence, IBM`

17 `Chris Kaler, Microsoft`

18 **Editors:**

19 `Thomas DeMartini, ContentGuard, Inc.`

20 `Anthony Nadalin, IBM`

21 `Chris Kaler, Microsoft`

22 `Ronald Monzillo, Sun`

23 `Phillip Hallam-Baker, Verisign`

24 **Abstract:**

25 `This document describes how to use ISO/IEC 21000-5 Rights Expressions with the Web`
26 `Services Security (WSS) specification.`

27
28
29
30
31
32
33
34
35
36
37
38

Status:

The status of this document is Committee Specification. Please send comments to the editors.

If you are on the wss@lists.oasis-open.org list for committee members, send comments there. If you are not on that list, subscribe to the wss-comment@lists.oasis-open.org list and send comments there. To subscribe, send an email message to wss-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

For patent disclosure information that may be essential to the implementation of this specification, and any offers of licensing terms, refer to the Intellectual Property Rights section of the OASIS Web Services Security Technical Committee (WSS TC) web page at <http://www.oasis-open.org/committees/wss/ipr.php>. General OASIS IPR information can be found at <http://www.oasis-open.org/who/intellectualproperty.shtml>.

Notices

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

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

52 Copyright © OASIS Open 2002-2005. All Rights Reserved.

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

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

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

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

Table of Contents

72	1	Introduction (Informative).....	5
73	2	Notations and Terminology (Normative).....	6
74	2.1	Notational Conventions.....	6
75	2.2	Namespaces.....	6
76	2.3	Terminology.....	7
77	3	Usage (Normative).....	8
78	3.1	Token Types.....	8
79	3.2	Processing Model.....	8
80	3.3	Attaching Security Tokens.....	8
81	3.4	Identifying and Referencing Security Tokens.....	8
82	3.5	Authentication.....	12
83	3.5.1	<r:keyHolder> Principal.....	12
84	3.6	Confidentiality.....	14
85	3.6.1	<r:keyHolder> Principal.....	15
86	3.7	Error Codes.....	16
87	4	Types of Licenses (Informative).....	17
88	4.1	Attribute Licenses.....	17
89	4.2	Sender Authorization.....	18
90	4.3	Issuer Authorization.....	18
91	5	Threat Model and Countermeasures (Informative).....	21
92	5.1	Eavesdropping.....	21
93	5.2	Replay.....	21
94	5.3	Message Insertion.....	22
95	5.4	Message Deletion.....	22
96	5.5	Message Modification.....	22
97	5.6	Man-in-the-Middle.....	22
98	6	References.....	23
99		Appendix A: Acknowledgements.....	24
100		Appendix B: Revision History.....	27
101			

102

1 Introduction (Informative)

103 The Web Services Security: SOAP Message Security [WS-Security] specification proposes a
104 standard set of SOAP extensions that can be used when building secure Web services to
105 implement message level integrity and confidentiality. This specification describes the use of
106 ISO/IEC 21000-5 Rights Expressions with respect to the WS-Security specification.

107 **2 Notations and Terminology (Normative)**

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

109 **2.1 Notational Conventions**

110 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",
111 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be
112 interpreted as described in [KEYWORDS].

113 Namespace URIs (of the general form "some-URI") represent some application-dependent or
114 context-dependent URI as defined in [URI].

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

119 **2.2 Namespaces**

120 The following namespaces are used in this document:

121

Prefix	Namespace
S	http://www.w3.org/2003/05/soap-envelope
ds	http://www.w3.org/2000/09/xmldsig#
xenc	http://www.w3.org/2001/04/xmlenc#
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
wsse11	http://docs.oasis-open.org/wss/oasis-wss-wssecurity-secext-1.1.xsd
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd
r	urn:mpeg:mpeg21:2003:01-REL-R-NS

sx	urn:mpeg:mpeg21:2003:01-REL-SX-NS
----	-----------------------------------

122

Table 1 Namespace Prefixes

123

124 **2.3 Terminology**

125 This specification employs the terminology defined in the Web Services Security: SOAP Message
126 Security [WS-Security] Specification.

127 Defined below are the basic definitions for additional terminology used in this specification.

128 **License** – ISO/IEC 21000-5 Rights Expression

129 3 Usage (Normative)

130 This section describes the syntax and processing rules for the use of licenses with
131 the Web Services Security: Soap Message Security specification [WS-Security].

132 3.1 Token Types

133 When a URI value is used to indicate a license according to this profile, its value MUST be
134 <http://docs.oasis-open.org/wss/oasis-wss-rel-token-profile-1.0.pdf#license>.

135 Note: This URI is for both the ValueType and TokenType attributes. It is also for use by any
136 elements or attributes that require a token type URI and are defined in another specification
137 taking advantage of REL Tokens.

138 3.2 Processing Model

139 The processing model for WS-Security with licenses is no different from that of WS-Security with
140 other token formats as described in Web Services Security: SOAP Message Security [WS-
141 Security].

142 At the token level, a processor of licenses MUST conform to the required validation and
143 processing rules defined in ISO/IEC 21000-5 [REL].

144 3.3 Attaching Security Tokens

145 Licenses are attached to SOAP messages using WS-Security by placing the license
146 element inside the <wsse:Security> header. The following example illustrates a
147 SOAP message with a license.

```
148 <S:Envelope xmlns:S="...">  
149   <S:Header>  
150     <wsse:Security xmlns:wsse="...">  
151       <r:license xmlns:r="...">  
152         ...  
153       </r:license>  
154       ...  
155     </wsse:Security>  
156   </S:Header>  
157   <S:Body>  
158     ...  
159   </S:Body>  
160 </S:Envelope>
```

161 3.4 Identifying and Referencing Security Tokens

162 The Web Services Security: SOAP Message Security [WS-Security] specification defines the
163 *wsu:id* attribute as the common mechanism for identifying security tokens (the specification

164 describes the reasons for this). Licenses have an additional identification mechanism available:
 165 their licenseld attribute, the value of which is a URI. The following example shows a license that
 166 uses both mechanisms:

```

167 <r:license xmlns:r="..." xmlns:wsu="..."
168   licenseId="urn:foo:SecurityToken:ef375268"
169   wsu:Id="SecurityToken-ef375268">
170   ...
171 </r:license>
  
```

172 Licenses can be referenced either according to their location or their licenseld. Location
 173 references are dependent on location and can be either local or remote. Licenseld references
 174 are not dependent on location.

175 Local location references are RECOMMENDED when they can be used. Remote location
 176 references are OPTIONAL for cases where it is not feasible to transmit licenses with the SOAP
 177 message. Licenseld references are OPTIONAL for cases where location is unknown or cannot
 178 be indicated.

179 WS-Security specifies that tokens are referenced using the <wsse:SecurityTokenReference>
 180 element.

181 Implementations compliant with this profile SHOULD set the
 182 /wsse:SecurityTokenReference/wsse:Reference/@ValueType attribute to http://docs.oasis-
 183 open.org/wss/oasis-wss-rel-token-profile-1.0.pdf#license when using
 184 wsse:SecurityTokenReference to refer to a license by licenseld. This is OPTIONAL when
 185 referring to a license by location.

186 The following table demonstrates the use of the <wsse:SecurityTokenReference> element to
 187 refer to licenses.

By Location	Local	<pre> <wsse:SecurityTokenReference> <wsse:Reference URI="#SecurityToken-ef375268" /> </wsse:SecurityTokenReference> </pre>
	Remote	<pre> <wsse:SecurityTokenReference> <wsse:Reference URI="http://www.foo.com/ef375268.xml" /> </wsse:SecurityTokenReference> </pre>
By licenseld		<pre> <wsse:SecurityTokenReference> <wsse:Reference URI="urn:foo:SecurityToken:ef375268" ValueType="http://docs.oasis- open.org/wss/oasis-wss-rel-token-profile- 1.0.pdf#license" /> </wsse:SecurityTokenReference> </pre>

188 **Table 2. <wsse:SecurityTokenReference>**

189 The following example demonstrates how a <wsse:SecurityTokenReference> can be used to
190 indicate that the message parts specified inside the <ds:SignedInfo> element were signed using
191 a key from the license referenced by licenseld in the <ds:KeyInfo> element.

```
192 <S:Envelope xmlns:S="..." xmlns:ds="...">  
193   <S:Header>  
194     <wsse:Security xmlns:wsse="...">  
195       <r:license xmlns:r="..."  
196         licenseId="urn:foo:SecurityToken:ef375268" xmlns:wsu="..."  
197         wsu:Id="SecurityToken-ef375268">  
198         ...  
199       </r:license>  
200       ...  
201     <ds:Signature>  
202       <ds:SignedInfo>  
203         ...  
204       </ds:SignedInfo>  
205       <ds:SignatureValue>...</ds:SignatureValue>  
206       <ds:KeyInfo>  
207         <wsse:SecurityTokenReference>  
208           <wsse:Reference  
209             URI="#SecurityToken-ef375268"  
210           />  
211         </wsse:SecurityTokenReference>  
212       </ds:KeyInfo>  
213     </ds:Signature>  
214   </wsse:Security>  
215 </S:Header>  
216 <S:Body>  
217   ...  
218 </S:Body>  
219 </S:Envelope>
```

220 The following example shows a signature over a local license using a location reference to that
221 license. The example demonstrates how the integrity of an (unsigned) license can be preserved
222 by signing it in the <wsse:Security> header.

```
223 <S:Envelope xmlns:S="..." xmlns:wsu="..." >  
224   <S:Header>  
225     <wsse:Security xmlns:wsse="...">  
226       <r:license xmlns:r="..." wsu:Id="SecurityToken-ef375268">  
227         ...  
228       </r:license>  
229       ...  
230     <wsse:SecurityTokenReference wsu:Id="Str1">  
231       <wsse:Reference  
232         URI="#SecurityToken-ef375268"  
233       />  
234     </wsse:SecurityTokenReference>  
235     ...  
236   <ds:Signature>  
237     <ds:SignedInfo>  
238       ...  
239     <ds:Reference URI="#Str1">  
240       <ds:Transforms>  
241         <ds:Transform
```

```

242           Algorithm="http://schemas.xmlsoap.org/2003/06/STR-
243 Transform">
244           <ds:CanonicalizationMethod
245             Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
246 20010315"/>
247           </ds:Transform>
248         </ds:Transforms>
249         <ds:DigestMethod
250           Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
251         />
252         <ds:DigestValue>...</ds:DigestValue>
253       </ds:Reference>
254     </ds:SignedInfo>
255     <ds:SignatureValue>...</ds:SignatureValue>
256     <ds:KeyInfo>...</ds:KeyInfo>
257   </ds:Signature>
258 </wsse:Security>
259 </S:Header>
260 <S:Body>
261   ...
262 </S:Body>
263 </S:Envelope>

```

264 Note: since licenses allow the use of the wsu:Id attribute, it is usually not necessary to use the
265 STR-Transform because the license can be referred to directly in the ds:SignedInfo as shown in
266 the following example:

```

267 <S:Envelope xmlns:S="..." xmlns:ds="...">
268   <S:Header>
269     <wsse:Security xmlns:wsse="...">
270       <r:license xmlns:r="..." xmlns:wsu="..." wsu:Id="SecurityToken-
271 ef375268">
272         ...
273       </r:license>
274       ...
275     <ds:Signature>
276       <ds:SignedInfo>
277         ...
278         <ds:Reference URI="#SecurityToken-ef375268">
279           <ds:DigestMethod
280             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
281           />
282           <ds:DigestValue>...</ds:DigestValue>
283         </ds:Reference>
284       </ds:SignedInfo>
285       <ds:SignatureValue>...</ds:SignatureValue>
286       <ds:KeyInfo>...</ds:KeyInfo>
287     </ds:Signature>
288   </wsse:Security>
289 </S:Header>
290 <S:Body>
291   ...
292 </S:Body>
293 </S:Envelope>

```

294 3.5 Authentication

295 The Web Services Security: SOAP Message Security [WS-Security] specification does not dictate
296 how claim confirmation must be performed. As well, the REL allows for multiple types of
297 confirmation. This profile of WS-Security REQUIRES that message senders and receivers
298 support claim confirmation for <r:keyHolder> principals. It is RECOMMENDED that an XML
299 Signature be used to establish the relationship between the message sender and the claims. This
300 is especially RECOMMENDED whenever the SOAP message exchange is conducted over an
301 unprotected transport.

302 The following table enumerates the mandatory principals to be supported by claim confirmation
303 and summarizes their associated processing models. It should be noted that this table is not all-
304 encompassing, and it is envisioned that future specifications may expand this table over time.

Principal	RECOMMENDED Processing Rules
<r:keyHolder>	The message sender adds (to the security header) an XML Signature that can be verified with the key information specified in the <r:keyHolder> of the referenced license.

305 **Table 3. Processing Rules for Claim Confirmation**

306 Note that the high-level processing model described in the following sections does not
307 differentiate between message author and message sender as would be necessary to guard
308 against replay attacks. The high-level processing model also does not take into account
309 requirements for authentication of receiver by sender or for message or token confidentiality.
310 These concerns must be addressed by means other than those described in the high-level
311 processing model. If confidentiality of the token in the message is important, then use the
312 approach defined by [WS-Security] to encrypt the token.

313 3.5.1 <r:keyHolder> Principal

314 The following sections describe the <r:keyHolder> method of establishing the correspondence
315 between a SOAP message sender and the claims within a license.

316 **Sender**

317 The message sender MUST include within the <wsse:Security> header element a <r:license>
318 containing at least one <r:grant> to an <r:keyHolder> identifying the key to be used to confirm the
319 claims. If the message sender includes an <r:license> containing more than one <r:grant> to an
320 <r:keyHolder>, then all of those <r:keyHolder> elements MUST be equal.

321 In order for the receiver to perform claim confirmation, the sender MUST demonstrate knowledge
322 of the confirmation key. The sender MAY accomplish this by using the confirmation key to sign
323 content from within the message and by including the resulting <ds:Signature> element in the
324 <wsse:Security> header element. <ds:Signature> elements produced for this purpose MUST

325 conform to the canonicalization and token inclusion rules defined in the core WS-Security
326 specification and this profile specification.

327 Licenses that contain at least one <r:grant> to an <r:keyHolder> SHOULD contain an <r:issuer>
328 with a <ds:Signature> element that identifies the license issuer to the relying party and protects
329 the integrity of the confirmation key established by the license issuer.

330 Receiver

331 If the receiver determines that the sender has demonstrated knowledge of a confirmation key as
332 specified in an <r:keyHolder>, then the claims (found in the licenses) pertaining to that
333 <r:keyHolder> MAY be attributed to the sender. If one of these claims is an identity and if the
334 conditions of that claim are satisfied, then any elements of the message whose integrity is
335 protected by the confirmation key MAY be considered to have been authored by that identity.

336 Example

337 The following example illustrates how a license security token having an <r:keyHolder> principal
338 can be used with a <ds:Signature> to establish that John Doe is requesting a stock report on
339 FOO.

```
340 <S:Envelope xmlns:S="...">
341   <S:Header>
342     <wsse:Security xmlns:wsse="...">
343       <r:license xmlns:r="..."
344 licenseId="urn:foo:SecurityToken:ef375268">
345         <r:grant>
346           <r:keyHolder>
347             <r:info>
348               <ds:KeyValue>...</ds:KeyValue>
349             </r:info>
350           </r:keyHolder>
351           <r:possessProperty/>
352           <sx:commonName xmlns:sx="...">John Doe</sx:commonName>
353         </r:grant>
354         <r:issuer>
355           <ds:Signature>...</ds:Signature>
356         </r:issuer>
357       </r:license>
358     </S:Header>
359     <ds:Signature>
360       <ds:SignedInfo>
361         ...
362         <ds:Reference URI="#MsgBody">
363           <ds:DigestMethod
364             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
365           />
366           <ds:DigestValue>...</ds:DigestValue>
367         </ds:Reference>
368       </ds:SignedInfo>
369       <ds:SignatureValue>...</ds:SignatureValue>
370     </ds:Signature>
371   </S:Envelope>
```

373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392

```

    <wsse:SecurityTokenReference>
      <wsse:Reference
        URI="urn:foo:SecurityToken:ef375268"
        ValueType="http://docs.oasis-open.org/wss/oasis-wss-rel-
token-profile-1.0.pdf#license"
      />
    </wsse:SecurityTokenReference>
  </ds:KeyInfo>
</ds:Signature>

</wsse:Security>
</S:Header>

<S:Body wsu:Id="MsgBody" xmlns:wsu="...">
  <ReportRequest>
    <TickerSymbol>FOO</TickerSymbol>
  </ReportRequest>
</S:Body>
</S:Envelope>

```

393 **3.6 Confidentiality**

394 This section details how licenses may be used to protect the confidentiality of a SOAP message
395 within WS-Security. The Web Services Security: SOAP Message Security [WS-Security]
396 specification does not dictate how confidentiality must be performed. As well, the REL allows for
397 multiple types of confidentiality. This profile of WS-Security REQUIRES that message senders
398 and receivers support confidentiality for <r:keyHolder> principals. It is RECOMMENDED that
399 XML Encryption be used to ensure confidentiality. This is especially RECOMMENDED whenever
400 the SOAP message exchange is conducted over an unprotected transport.

401 The following table enumerates the mandatory principals to be supported for confidentiality and
402 summarizes their associated processing models. It should be noted that this table is not all-
403 encompassing, and it is envisioned that future specifications may expand this table over time.

Principal	RECOMMENDED Processing Rules
<r:keyHolder>	The message sender adds (to the security header) either 1) an <xenc:ReferenceList> that points to one or more <xenc:EncryptedData> elements that can be decrypted with a key which can be determined from information specified in the <r:keyHolder> of the referenced license or 2) an <xenc:EncryptedKey> that can be decrypted with a key determined from information specified in the <r:keyHolder> of the referenced license.

404 **Table 4. Processing Rules for Confidentiality**

405 Note that this section deals only with Confidentiality. Details of authentication of the sender by
406 the receiver must be addressed by means other than those described in this section (see the
407 previous section).

408 3.6.1 <r:keyHolder> Principal

409 The following sections describe the <r:keyHolder> method of establishing confidentiality using a
410 license.

411 Sender

412 The message sender MUST include within the <wsse:Security> header element a <r:license>
413 containing at least one <r:grant> to an <r:keyHolder> identifying the key used to encrypt some
414 data or key. If the message sender includes an <r:license> containing more than one <r:grant> to
415 an <r:keyHolder>, then all of those <r:keyHolder> elements MUST be equal.

416 In order for the receiver to know when to decrypt the data or key, the sender MUST indicate the
417 encryption in the message. The sender MAY accomplish this by placing an
418 <xenc:EncryptedData> or <xenc:EncryptedKey> in the appropriate place in the message and by
419 including the resulting <xenc:ReferenceList> or <xenc:EncryptedKey> element in the
420 <wsse:Security> header element. <xenc:ReferenceList> or <xenc:EncryptedKey> elements
421 produced for this purpose MUST conform to the rules defined in the core WS-Security
422 specification and this profile specification.

423 Receiver

424 If the receiver determines that he has knowledge of a decryption key as specified in an
425 <r:keyHolder>, then he MAY decrypt the associated data or key. In the case of decrypting a key,
426 he may then recursively decrypt any data or key that that key can decrypt.

427

428 Example

429 The following example illustrates how a license containing a <r:keyHolder> principal can be used
430 with XML encryption schema elements to protect the confidentiality of a message using a
431 separate encryption key given in the <xenc:EncryptedKey> in the security header.

432 In this example, the r:license element provides information about the recipient's RSA public key
433 (i.e., KeyValue in keyHolder) used to encrypt the symmetric key carried in the EncryptedKey
434 element. The recipient uses this information to determine the correct private key to use in
435 decrypting the symmetric key. The symmetric key is then used to decrypt the EncryptedData child
436 of the Body element.

437

```
438 <S:Envelope xmlns:S="..." xmlns:ds="...">  
439 <S:Header>  
440 <wsse:Security xmlns:wsse="...">  
441 <r:license xmlns:r="..."  
442 licenseId="urn:foo:SecurityToken:ef375268">
```

```

443     <r:grant>
444         <r:keyHolder>
445             <r:info>
446                 <ds:KeyValue>...</ds:KeyValue>
447             </r:info>
448         </r:keyHolder>
449         <r:possessProperty/>
450         <sx:commonName xmlns:sx="...">SOME COMPANY</sx:commonName>
451     </r:grant>
452     <r:issuer>
453         <ds:Signature>...</ds:Signature>
454     </r:issuer>
455 </r:license>
456 <xenc:EncryptedKey xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
457     <xenc:EncryptionMethod
458         Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
459     <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
460         <wsse:SecurityTokenReference>
461             <wsse:Reference URI="urn:foo:SecurityToken:ef375268"/>
462         </wsse:SecurityTokenReference>
463     </KeyInfo>
464     <xenc:CipherData>
465         <xenc:CipherValue>dNYS...fQ=</xenc:CipherValue>
466     </xenc:CipherData>
467     <xenc:ReferenceList>
468         <xenc:DataReference URI="#enc"/>
469     </xenc:ReferenceList>
470 </xenc:EncryptedKey>
471 </wsse:Security>
472 </S:Header>
473 <S:Body wsu:Id="body"
474     xmlns:wsu="http://schemas.xmlsoap.org/ws/2003/06/utility">
475     <xenc:EncryptedData Id="enc"
476         Type="http://www.w3.org/2001/04/xmlenc#Content"
477         xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
478         <xenc:EncryptionMethod
479             Algorithm="http://www.w3.org/2001/04/xmlenc#tripleDES-cbc"/>
480         <xenc:CipherData>
481             <xenc:CipherValue>d2s...GQ=</xenc:CipherValue>
482         </xenc:CipherData>
483     </xenc:EncryptedData>
484 </S:Body>
485 </S:Envelope>

```

486 3.7 Error Codes

487 It is RECOMMENDED that the error codes defined in the Web Services Security:
488 SOAP Message Security [WS-Security] specification are used. However,
489 implementations MAY use custom errors, defined in private namespaces if they
490 desire. Care should be taken not to introduce security vulnerabilities in the errors
491 returned.

492

4 Types of Licenses (Informative)

493

4.1 Attribute Licenses

494

In addition to key information, licenses can carry information about attributes of those keys.

495

Examples of such information on a client are e-mail address or common name. A service's key,

496

on the other hand, might be associated with a DNS name and common name.

497

The following is an example client attribute license.

498

```
<r:license xmlns:r="..." xmlns:ds="..."
licenseId="urn:foo:SecurityToken:ef375268">
  <r:inventory>
    <r:keyHolder licensePartId="client">
      <r:info>
        <ds:KeyValue>FDFEWEFF...</ds:KeyValue>
      </r:info>
    </r:keyHolder>
  </r:inventory>
  <r:grant>
    <r:keyHolder licensePartIdRef="client"/>
    <r:possessProperty/>
    <sx:commonName>John Doe</sx:commonName>
  </r:grant>
  <r:grant>
    <r:keyHolder licensePartIdRef="client"/>
    <r:possessProperty/>
    <sx:emailName>jd@foo.com</sx:emailName>
  </r:grant>
  <r:issuer>
    <ds:Signature>...</ds:Signature>
  </r:issuer>
</r:license>
```

521

The following is an example service attribute license.

522

```
<r:license xmlns:r="..." xmlns:ds="..."
licenseId="urn:foo:SecurityToken:ef375268">
  <r:inventory>
    <r:keyHolder licensePartId="service">
      <r:info>
        <ds:KeyValue>FDFEWEFF...</ds:KeyValue>
      </r:info>
    </r:keyHolder>
  </r:inventory>
  <r:grant>
    <r:keyHolder licensePartIdRef="service"/>
    <r:possessProperty/>
    <sx:commonName>MyService Company</sx:commonName>
  </r:grant>
  <r:grant>
    <r:keyHolder licensePartIdRef="service"/>
    <r:possessProperty/>
    <sx:dnsName>www.myservice.com</sx:dnsName>
  </r:grant>
  <r:issuer>
    <ds:Signature>...</ds:Signature>
  </r:issuer>
```

543

544 </r:license>

545 Additional examples of and processing rules for the use of attribute licenses can be found in the
546 above sections on Authentication and Confidentiality.

547 4.2 Sender Authorization

548 Licenses may be used by a sender as proof of authorization to perform a certain action on a
549 particular resource. This WS-Security specification does not describe how authorization must be
550 performed. In the context of web services, a sender can send to a receiver an authorization
551 license in the security header as proof of authorization to call the sender. Typically, this
552 authorization license is signed by a trusted authority and conforms to the syntax pattern specified
553 below.

```
554 <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">  
555   <r:grant>  
556     <r:keyHolder>  
557       <r:info>  
558         <ds:KeyValue>FDFEWEFF...</ds:KeyValue>  
559       </r:info>  
560     </r:keyHolder>  
561     <sx:rightUri definition='...' />  
562     <x:someResource />  
563     <x:someCondition />  
564   </r:grant>  
565   <r:issuer>  
566     <ds:Signature>...</ds:Signature>  
567   </r:issuer>  
568 </r:license>
```

569 The above license contains an authorization grant authorizing the keyholder (sender's public
570 key), the right to exercise the right identified in the <sx:rightUri> element. The resource in the
571 license typically corresponds to the semantics of the URI given in the definition attribute of the
572 <sx:rightUri> element. The entire license along with the <ds:Signature> element in the <r:issuer>
573 certifies the fact that the principal (<keyholder>) is granted the authorization to exercise the right
574 in the <sx:rightUri> element over the specified resource. The integrity of the license is usually
575 protected with a digital signature contained within the <ds:Signature>.

576 4.3 Issuer Authorization

577 To enunciate that a particular issuer is allowed to issue particular types of licenses, one can use
578 the kind of license described here. Issuer authorization licenses can accompany other licenses in
579 the security header such as those used for authentication, sender authorization, or other issuer
580 authorizations. These issuer authorization licenses might help complete the authorization proof
581 that is required for authorizing or authenticating a particular sender.

582

583 The following license is an example issuer authorization license for authorizing an issuer to issue
584 a simple attribute license.

```
585 <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">  
586   <r:grant>  
587     <r:forAll varName='K' />  
588     <r:forAll varName='P' />  
589     <r:keyHolder>  
590       <r:info>
```

```

591         <ds:KeyValue>FDFEWEFF...</ds:KeyValue>
592     </r:info>
593 </r:keyHolder>
594 <r:issue/>
595 <r:grant>
596     <r:keyHolder varRef='K' />
597     <r:possessProperty/>
598     <r:propertyAbstract varRef='P' />
599 </r:grant>
600 </r:grant>
601 <r:issuer>
602     <ds:Signature>...</ds:Signature>
603 </r:issuer>
604 </r:license>

```

605 The following license is an example issuer authorization license for authorizing an issuer to issue
606 sender authorization licenses.

```

607 <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
608     <r:grant>
609         <r:forAll varName='K' />
610         <r:forAll varName='R' />
611         <r:keyHolder>
612             <r:info>
613                 <ds:KeyValue>FDFEWEFF...</ds:KeyValue>
614             </r:info>
615         </r:keyHolder>
616         <r:issue/>
617         <r:grant>
618             <r:keyHolder varRef='K' />
619             <sx:rightUri definition='...' />
620             <r:resource varRef='R' />
621         </r:grant>
622     </r:grant>
623     <r:issuer>
624         <ds:Signature>...</ds:Signature>
625     </r:issuer>
626 </r:license>

```

627 The following license is an example issuer authorization license for authorizing an issuer to issue
628 (to other issuers) issuer authorization licenses allowing those other issuers to issue simple
629 attribute licenses, such as those that can be used for authentication or confidentiality.

```

630 <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
631     <r:grant>
632         <r:forAll varName='I' />
633         <r:keyHolder>
634             <r:info>
635                 <ds:KeyValue>FDFEWEFF...</ds:KeyValue>
636             </r:info>
637         </r:keyHolder>
638         <r:issue/>
639         <r:grant>
640             <r:forAll varName='K' />
641             <r:forAll varName='P' />
642             <r:keyHolder varRef='I' />
643             <r:issue/>
644             <r:grant>
645                 <r:keyHolder varRef='K' />
646                 <r:possessProperty/>
647                 <r:propertyAbstract varRef='P' />
648             </r:grant>
649         </r:grant>
650     </r:grant>
651     <r:issuer>

```

652
653
654
655

```
<ds:Signature>...</ds:Signature>  
</r:issuer>  
</r:license>
```

656

5 Threat Model and Countermeasures (Informative)

657

658 This section addresses the potential threats that a SOAP message may encounter and the
659 countermeasures that may be taken to thwart such threats. A SOAP message containing licenses
660 may face threats in various contexts. This includes the cases where the message is in transit,
661 being routed through a number of intermediaries, or during the period when the message is in
662 storage.

663 The use of licenses with WS-Security introduces no new threats beyond those identified for the
664 REL or WS-Security with other types of security tokens. Message alteration and eavesdropping
665 can be addressed by using the integrity and confidentiality mechanisms described in WS-
666 Security. Replay attacks can be addressed by using of message timestamps and caching, as well
667 as other application-specific tracking mechanisms. For licenses, ownership is verified by the use
668 of keys; man-in-the-middle attacks are generally mitigated. It is strongly RECOMMENDED that all
669 relevant and immutable message data be signed. It should be noted that transport-level security
670 MAY be used to protect the message and the security token. In order to trust licenses, they
671 SHOULD be signed natively and/or using the mechanisms outlined in WS-Security. This allows
672 readers of the licenses to be certain that the licenses have not been forged or altered in any way.
673 It is strongly RECOMMENDED that the <r:license> elements be signed (either within the token,
674 as part of the message, or both).

675 The following few sections elaborate on the afore-mentioned threats and suggest
676 countermeasures.

677

5.1 Eavesdropping

678 Eavesdropping is a threat to the confidentiality of the message, and is common to all types of
679 network protocols. The routing of SOAP messages through intermediaries increases the potential
680 incidences of eavesdropping. Additional opportunities for eavesdropping exist when SOAP
681 messages are persisted.

682 To provide maximum protection from eavesdropping, licenses, license references, and sensitive
683 message content SHOULD be encrypted such that only the intended audiences can view their
684 content. This removes threats of eavesdropping in transit, but does not remove risks associated
685 with storage or poor handling by the receiver.

686 Transport-layer security MAY be used to protect the message from eavesdropping while in
687 transport, but message content must be encrypted above the transport if it is to be protected from
688 eavesdropping by intermediaries.

689

5.2 Replay

690 The reliance on authority protected (e.g. signed) licenses to <r:keyHolder> principals precludes
691 all but the key holder from binding the licenses to a SOAP message. Although this mechanism

692 effectively restricts message authorship to the holder of the confirmation key, it does not preclude
693 the capture and resubmission of the message by other parties.

694 Replay attacks can be addressed by using message timestamps and caching, as well as other
695 application-specific tracking mechanisms.

696 **5.3 Message Insertion**

697 This profile of WS-Security is not vulnerable to message insertion attacks. Higher-level protocols
698 built on top of SOAP and WS-Security should avoid introducing message insertion threats and
699 provide proper countermeasures for any they do introduce.

700 **5.4 Message Deletion**

701 This profile of WS-Security is not vulnerable to message deletion attacks other than denial of
702 service. Higher-level protocols built on top of SOAP and WS-Security should avoid introducing
703 message deletion threats and provide proper countermeasures for any they do introduce.

704 **5.5 Message Modification**

705 Message Modification poses a threat to the integrity of a message. The threat of message
706 modification can be thwarted by signing the relevant and immutable content by the key holder.
707 The receivers SHOULD only trust the integrity of those segments of the message that are signed
708 by the key holder.

709 To ensure that message receivers can have confidence that received licenses have not been
710 forged or altered since their issuance, licenses appearing in <wsse:Security> header elements
711 SHOULD be integrity protected (e.g. signed) by their issuing authority. It is strongly
712 RECOMMENDED that a message sender sign any <r:license> elements that it is confirming and
713 that are not signed by their issuing authority.

714 Transport-layer security MAY be used to protect the message and contained licenses and/or
715 license references from modification while in transport, but signatures are required to extend such
716 protection through intermediaries.

717 **5.6 Man-in-the-Middle**

718 This profile of WS-Security is not vulnerable to man-in-the-middle attacks. Higher-level protocols
719 built on top of SOAP and WS-Security should avoid introducing Man-in-the-Middle threats and
720 provide proper countermeasures for any they do introduce.

721

722

6 References

- 723 **[KEYWORDS]** S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels,"
724 RFC 2119, Harvard University, March 1997,
725 <http://www.ietf.org/rfc/rfc2119.txt>
- 726 **[REL]** ISO/IEC 21000-5:2004, "Information technology -- Multimedia framework
727 (MPEG-21) -- Part 5: Rights Expression Language,"
728 <http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=36095&ICS1=35&ICS2=40&ICS3=>
729
- 730 **[SOAP]** D. Box, D Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H.
731 Frystyk Nielsen, S Thatte, D. Winer. Simple Object Access Protocol
732 (SOAP) 1.1, W3C Note 08 May 2000, <http://www.w3.org/TR/SOAP/>
733
734 W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging
735 Framework", 23 June 2003
- 736 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers
737 (URI): Generic Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox
738 Corporation, August 1998, <http://www.ietf.org/rfc/rfc2396.txt>
739
740 T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers
741 (URI): Generic Syntax," RFC 3986, MIT/LCS, Day Software, Adobe
742 Systems, January 2005, <http://www.ietf.org/rfc/rfc3986.txt>.
- 743 **[WS-Security]** OASIS Standard 200401, "Web Services Security: Soap Message
744 Security 1.0 (WS-Security 2004)," March 2004, [http://docs.oasis-](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)
745 [open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)
746
747 OASIS Standard, "Web Services Security: Soap Message Security 1.1
748 (WS-Security 2004)," November 2005, [http://docs.oasis-open.org/wss/](http://docs.oasis-open.org/wss/oasis-wss-soap-message-security-1.1.pdf)
749 [oasis-wss-soap-message-security-1.1.pdf](http://docs.oasis-open.org/wss/oasis-wss-soap-message-security-1.1.pdf)
- 750 **[XML-ns]** T. Bray, D. Hollander, A. Layman. Namespaces in XML. W3C
751 Recommendation. January 1999, [http://www.w3.org/TR/1999/REC-xml-](http://www.w3.org/TR/1999/REC-xml-names-19990114)
752 [names-19990114](http://www.w3.org/TR/1999/REC-xml-names-19990114)
- 753 **[XML Signature]** D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. XML-
754 Signature Syntax and Processing, W3C Recommendation, 12 February
755 2002.
- 756

Appendix A: Acknowledgements

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
Mark	Hayes	formerly of VeriSign
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
Paul	Cotton	Microsoft
Vijay	Gajjala	Microsoft
Martin	Gudgin	Microsoft
Chris	Kaler	Microsoft
Frederick	Hirsch	Nokia
Abbie	Barbir	Nortel
Vamsi	Motukuru	Oracle
Prateek	Mishra	Principal Identity
Ben	Hammond	RSA Security
Rob	Philpott	RSA Security
Blake	Dournaee	Sarvega
Sundeep	Peechu	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

759 **Previous Contributors:**

Peter	Dapkus	BEA
Guillermo	Lao	ContentGuard
TJ	Pannu	ContentGuard
Xin	Wang	ContentGuard
Shawn	Sharp	Cyclone Commerce
Ganesh	Vaideeswaran	Documentum
John	Hughes	Entegrity
Tim	Moses	Entrust
Carolina	Canales-Valenzuela	Ericsson
Davanum	Srinivas	formerly of Computer Associates
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
Bob	Morgan	Individual/Internet2
Kate	Cherry	Lockheed Martin
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/Sun
Senthil	Sengodan	Nokia
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Vipin	Samar	Oracle

Jerry
Eric
Stuart
Andrew
Peter
Martijn
Jonathan
Yassir
Michael
Don
Morten

Schwarz
Gravengaard
King
Nash
Rostin
de Boer
Tourzan
Elley
Nguyen
Adams
Jorgensen

Oracle
Reactivity
Reed Elsevier
RSA Security
RSA Security
SAP
Sony
Sun
The IDA of Singapore
TIBCO
Vordel

760

761

Appendix B: Revision History

Rev	Date	What

762

763