



Web Services Security Rights Expression Language (REL) Token Profile

18 June 2004

Document identifier:

urn:oasis:names:tc:WSS:1.0:profiles:REL

Location:

<http://docs.oasis-open.org/wss/2004/###/oasis-####-wss-REL-token-profile-1.0>

<http://www.oasis-open.org/committees/documents.php>

Editors:

Thomas	DeMartini	ContentGuard, Inc.
Anthony	Nadalin	IBM
Chris	Kaler	Microsoft Corporation
Ronald	Monzillo	Sun Microsystems
Phillip	Hallam-Baker	Verisign

Contributors:

Current voting members of the WSS TC (as of 01 June 2004)

Gene	Thurston	AmberPoint
Frank	Siebenlist	Argonne National Laboratory
Peter	Dapkus	BEA Systems, Inc.
Hal	Lockhart	BEA Systems, Inc.
Corinna	Witt	BEA Systems, Inc.
Merlin	Hughes	Betrusted (Baltimore Technologies)
Symon	Chang	CommerceOne
Davanum	Srinivas	Computer Associates
Thomas	DeMartini	ContentGuard, Inc.
Guillermo	Lao	ContentGuard, Inc.
TJ	Pannu	ContentGuard, Inc.
Sam	Wei	Documentum
John	Hughes	Entegrity
Tim	Moses	Entrust
Dana	Kaufman	Forum Systems, Inc.

Toshihiro	Nishimura	Fujitsu
Kefeng	Chen	GeoTrust
Irving	Reid	Hewlett-Packard
Yutaka	Kudo	Hitachi
Kojiro	Nakayama	Hitachi
Paula	Austel	IBM
Derek	Fu	IBM
Maryann	Hondo	IBM
Kelvin	Lawrence	IBM
Michael	McIntosh	IBM
Anthony	Nadalin	IBM
Nataraj	Nagaratnam	IBM
Ron	Williams	IBM
Don	Flinn	Individual
Bob	Morgan	Individual
Maneesh	Sahu	Individual
Kate	Cherry	Lockheed Martin
Paul	Cotton	Microsoft Corporation
Vijay	Gajjala	Microsoft Corporation
Alan	Geller	Microsoft Corporation
Chris	Kaler	Microsoft Corporation
Ellen	McDermott	Microsoft Corporation
John	Shewchuk	Microsoft Corporation
Richard	Levinson	Netegrity, Inc.
Prateek	Mishra	Netegrity, Inc.
Frederick	Hirsch	Nokia
Abbie	Barbir	Nortel Networks
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Steve	Anderson	OpenNetwork
Jerry	Schwarz	Oracle
Ramana	Turlapati	Oracle
Ben	Hammond	RSA Security
Andrew	Nash	RSA Security
Rob	Philpott	RSA Security
Eric	Gravengaard	Reactivity
Martijn	de Boer	SAP
Blake	Dournaee	Sarvega
Coumara	Radja	Sarvega
Pete	Wenzel	SeeBeyond Technology Corporation
Jeff	Hodges	Sun Microsystems
Ronald	Monzillo	Sun Microsystems
Jan	Alexander	Systinet
John	Weiland	US Dept of the Navy
Phillip	Hallam-Baker	Verisign

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

Xin	Wang	ContentGuard, Inc.
Hiroshi	Maruyama	IBM
Hemma	Prafullchandra	Verisign

16 **Abstract:**
17 This document describes how to use ISO/IEC 21000-5 Rights Expressions with the Web
18 Services Security: SOAP Message Security [WS-Security] specification.

19 **Status:**
20 This is a Committee Draft. Please send comments to the editors.
21 Committee members should send comments on this specification to the
22 <mailto:wss@lists.oasis-open.org> list. Others should subscribe to and send comments to
23 the wss-comment@lists.oasis-open.org list. To subscribe, visit
24 <http://lists.oasis-open.org/ob/adm.pl>.
25 For information on whether any patents have been disclosed that may be essential to
26 implementing this specification, and any offers of patent licensing terms, please refer to
27 the Intellectual Property Rights section of the Web Services Security TC web page
28 (<http://www.oasis-open.org/committees/wss/ipr.php>).

Table of Contents

30	1	Introduction (Informative).....	5
31	2	Notations and Terminology (Normative).....	6
32	2.1	Notational Conventions.....	6
33	2.2	Namespaces.....	6
34	2.3	Terminology.....	7
35	3	Usage (Normative).....	8
36	3.1	Token Types.....	8
37	3.2	Processing Model.....	8
38	3.3	Attaching Security Tokens.....	8
39	3.4	Identifying and Referencing Security Tokens.....	8
40	3.5	Authentication.....	11
41	3.5.1	<r:keyHolder> Principal.....	12
42	3.6	Confidentiality.....	14
43	3.6.1	<r:keyHolder> Principal.....	15
44	3.7	Error Codes.....	16
45	4	Types of Licenses (Informative).....	17
46	4.1	Attribute Licenses.....	17
47	4.2	Sender Authorization.....	18
48	4.3	Issuer Authorization.....	18
49	5	Threat Model and Countermeasures (Informative).....	21
50	5.1	Eavesdropping.....	21
51	5.2	Replay.....	21
52	5.3	Message Insertion.....	22
53	5.4	Message Deletion.....	22
54	5.5	Message Modification.....	22
55	5.6	Man-in-the-Middle.....	22
56	6	References.....	23
57		Appendix A: Revision History.....	24
58		Appendix B: Notices.....	25
59			

60 **1 Introduction (Informative)**

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

65 2 Notations and Terminology (Normative)

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

67 2.1 Notational Conventions

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

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

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

77 2.2 Namespaces

78 The XML namespace [XML-ns] URIs that MUST be used by implementations of this specification
79 are as follows (note that different elements in this specification are from different namespaces):

```
80     http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
81     wssecurity-secext-1.0.xsd  
82     http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-  
83     wssecurity-utility-1.0.xsd  
84     urn:mpeg:mpeg21:2003:01-REL-R-NS
```

85 The following namespaces are used in this document:

86

Prefix	Namespace
S	http://www.w3.org/2001/12/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
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

87

Table 1 Namespace Prefixes

88

89 **2.3 Terminology**

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

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

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

94 3 Usage (Normative)

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

97 3.1 Token Types

98 When a URI value is used to indicate a license according to this profile, its value MUST be
99 <http://docs.oasis-open.org/wss/2004/###oasis-#####-wss-REL-token-profile-1.0#license>.

100 3.2 Processing Model

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

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

106 3.3 Attaching Security Tokens

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

```
110 <S:Envelope xmlns:S="...">  
111   <S:Header>  
112     <wsse:Security xmlns:wsse="...">  
113       <r:license xmlns:r="...">  
114         ...  
115       </r:license>  
116       ...  
117     </wsse:Security>  
118   </S:Header>  
119   <S:Body>  
120     ...  
121   </S:Body>  
122 </S:Envelope>
```

123 3.4 Identifying and Referencing Security Tokens

124 The Web Services Security: SOAP Message Security [WS-Security] specification defines the
125 *wsu:id* attribute as the common mechanism for identifying security tokens (the specification
126 describes the reasons for this). Licenses have an additional identification mechanism available:
127 their *licenseId* attribute, the value of which is a URI. The following example shows a license that
128 uses both mechanisms:

129
130
131
132
133

```
<r:license xmlns:r="..." xmlns:wssu="..."
  licenseId="urn:foo:SecurityToken:ef375268"
  wssu:Id="SecurityToken-ef375268">
  ...
</r:license>
```

134 Licenses can be referenced either according to their location or their licenseld. Location
135 references are dependent on location and can be either local or remote. Licenseld references
136 are not dependent on location.

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

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

143 Implementations compliant with this profile SHOULD set the
144 /wsse:SecurityTokenReference/wsse:Reference/@ValueType attribute to http://docs.oasis-
145 open.org/wss/2004/##/oasis-####-wss-REL-token-profile-1.0#license when using
146 wsse:SecurityTokenReference to refer to a license by licenseld. This is OPTIONAL when
147 referring to a license by location.

148 The following table demonstrates the use of the <wsse:SecurityTokenReference> element to
149 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/2004/##/oasis-####-wss-REL-token- profile-1.0#license" /> </wsse:SecurityTokenReference></pre>

150

Table 2. <wsse:SecurityTokenReference>

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

154
155

```
<S:Envelope xmlns:S="...">
  <S:Header>
```

```

156     <wsse:Security xmlns:wsse="...">
157         <r:license xmlns:r="..."
158 licenseId="urn:foo:SecurityToken:ef375268" xmlns:wsu="..."
159 wsu:Id="SecurityToken-ef375268">
160             ...
161         </r:license>
162         ...
163         <ds:Signature>
164             <ds:SignedInfo>
165                 ...
166             </ds:SignedInfo>
167             <ds:SignatureValue>...</ds:SignatureValue>
168             <ds:KeyInfo>
169                 <wsse:SecurityTokenReference>
170                     <wsse:Reference
171                         URI="#SecurityToken-ef375268"
172                     />
173                 </wsse:SecurityTokenReference>
174             </ds:KeyInfo>
175         </ds:Signature>
176     </wsse:Security>
177 </S:Header>
178 <S:Body>
179     ...
180 </S:Body>
181 </S:Envelope>

```

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

```

185 <S:Envelope xmlns:S="...">
186   <S:Header>
187     <wsse:Security xmlns:wsse="..."
188       <r:license xmlns:r="..." xmlns:wsu="..." wsu:Id="SecurityToken-
189 ef375268">
190         ...
191       </r:license>
192       ...
193       <wsse:SecurityTokenReference wsu:Id="Str1">
194         <wsse:Reference
195           URI="#SecurityToken-ef375268"
196         />
197       </wsse:SecurityTokenReference>
198       ...
199       <ds:Signature>
200         <ds:SignedInfo>
201           ...
202           <Reference URI="#Str1">
203             <Transforms>
204               <ds:Transform
205                 Algorithm="http://schemas.xmlsoap.org/2003/06/STR-
206 Transform">
207                 <ds:CanonicalizationMethod
208                   Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
209 20010315"/>
210               </ds:Transform>

```

```

211         </ds:Transforms>
212         <ds:DigestMethod
213             Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
214         />
215         <ds:DigestValue>...</ds:DigestValue>
216     </ds:Reference>
217 </ds:SignedInfo>
218 <ds:SignatureValue>...</ds:SignatureValue>
219 <ds:KeyInfo>...</ds:KeyInfo>
220 </ds:Signature>
221 </wsse:Security>
222 </S:Header>
223 <S:Body>
224     ...
225 </S:Body>
226 </S:Envelope>

```

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

```

230 <S:Envelope xmlns:S="...">
231   <S:Header>
232     <wsse:Security xmlns:wsse="...">
233       <r:license xmlns:r="..." xmlns:wsu="..." wsu:Id="SecurityToken-
234 ef375268">
235         ...
236       </r:license>
237       ...
238       <ds:Signature>
239         <ds:SignedInfo>
240           ...
241           <ds:Reference URI="#SecurityToken-ef375268">
242             <ds:DigestMethod
243               Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
244             />
245             <ds:DigestValue>...</ds:DigestValue>
246           </ds:Reference>
247         </ds:SignedInfo>
248         <ds:SignatureValue>...</ds:SignatureValue>
249         <ds:KeyInfo>...</ds:KeyInfo>
250       </ds:Signature>
251     </wsse:Security>
252   </S:Header>
253   <S:Body>
254     ...
255   </S:Body>
256 </S:Envelope>

```

257 3.5 Authentication

258 The Web Services Security: SOAP Message Security [WS-Security] specification does not dictate
259 how claim confirmation must be performed. As well, the REL allows for multiple types of
260 confirmation. This profile of WS-Security REQUIRES that message senders and receivers
261 support claim confirmation for <r:keyHolder> principals. It is RECOMMENDED that an XML

262 Signature be used to establish the relationship between the message sender and the claims. This
263 is especially RECOMMENDED whenever the SOAP message exchange is conducted over an
264 unprotected transport.

265 The following table enumerates the mandatory principals to be supported by claim confirmation
266 and summarizes their associated processing models. It should be noted that this table is not all-
267 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.

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

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

276 **3.5.1 <r:keyHolder> Principal**

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

279 **Sender**

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

284 In order for the receiver to perform claim confirmation, the sender MUST demonstrate knowledge
285 of the confirmation key. The sender MAY accomplish this by using the confirmation key to sign
286 content from within the message and by including the resulting <ds:Signature> element in the
287 <wsse:Security> header element. <ds:Signature> elements produced for this purpose MUST
288 conform to the canonicalization and token inclusion rules defined in the core WS-Security
289 specification and this profile specification.

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

293 Receiver

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

299 Example

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

```
303 <S:Envelope xmlns:S="...">
304   <S:Header>
305     <wsse:Security xmlns:wsse="...">
306       <r:license xmlns:r="..."
307 licenseId="urn:foo:SecurityToken:ef375268">
308       <r:grant>
309         <r:keyHolder>
310           <r:info>
311             <ds:KeyValue>...</ds:KeyValue>
312           </r:info>
313         </r:keyHolder>
314         <r:possessProperty/>
315         <sx:commonName xmlns:sx="...">John Doe</sx:commonName>
316       </r:grant>
317       <r:issuer>
318         <ds:Signature>...</ds:Signature>
319       </r:issuer>
320     </r:license>
321   </S:Header>
322   <ds:Signature>
323     <ds:SignedInfo>
324       ...
325       <ds:Reference URI="#MsgBody">
326         <ds:DigestMethod
327           Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
328         />
329         <ds:DigestValue>...</ds:DigestValue>
330       </ds:Reference>
331     </ds:SignedInfo>
332     <ds:SignatureValue>...</ds:SignatureValue>
333     <ds:KeyInfo>
334       <wsse:SecurityTokenReference>
335         <wsse:Reference
336           URI="urn:foo:SecurityToken:ef375268"
337           ValueType="http://docs.oasis-open.org/wss/2004/##/oasis-
338 #####-wss-REL-token-profile-1.0#license"
339         />
340       </wsse:SecurityTokenReference>
341     </ds:KeyInfo>
```

344
345
346
347
348
349
350
351
352
353
354
355

```
</ds:Signature>

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

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

</S:Envelope>
```

356 **3.6 Confidentiality**

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

364 The following table enumerates the mandatory principals to be supported for confidentiality and
365 summarizes their associated processing models. It should be noted that this table is not all-
366 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.

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

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

371 3.6.1 <r:keyHolder> Principal

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

374 Sender

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

379 In order for the receiver to know when to decrypt the data or key, the sender MUST indicate the
380 encryption in the message. The sender MAY accomplish this by placing an
381 <xenc:EncryptedData> or <xenc:EncryptedKey> in the appropriate place in the message and by
382 including the resulting <xenc:ReferenceList> or <xenc:EncryptedKey> element in the
383 <wsse:Security> header element. <xenc:ReferenceList> or <xenc:EncryptedKey> elements
384 produced for this purpose MUST conform to the rules defined in the core WS-Security
385 specification and this profile specification.

386 Receiver

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

390

391 Example

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

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

400

```
401 <S:Envelope xmlns:S="...">  
402   <S:Header>  
403     <wsse:Security xmlns:wsse="...">  
404       <r:license xmlns:r="..."  
405         licenseId="urn:foo:SecurityToken:ef375268">  
406         <r:grant>  
407           <r:keyHolder>  
408             <r:info>  
409               <ds:KeyValue>...</ds:KeyValue>  
410             </r:info>  
411           </r:keyHolder>
```

```

412     <r:possessProperty/>
413     <sx:commonName xmlns:sx="...">SOME COMPANY</sx:commonName>
414 </r:grant>
415 <r:issuer>
416     <ds:Signature>...</ds:Signature>
417 </r:issuer>
418 </r:license>
419 <xenc:EncryptedKey xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
420     <xenc:EncryptionMethod
421         Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
422     <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
423         <wsse:SecurityTokenReference>
424             <wsse:Reference URI="urn:foo:SecurityToken:ef375268"/>
425         </wsse:SecurityTokenReference>
426     </KeyInfo>
427     <xenc:CipherData>
428         <xenc:CipherValue>dNYS...fQ=</xenc:CipherValue>
429     </xenc:CipherData>
430     <xenc:ReferenceList>
431         <xenc:DataReference URI="#enc"/>
432     </xenc:ReferenceList>
433 </xenc:EncryptedKey>
434 </wsse:Security>
435 </S:Header>
436 <S:Body wsu:Id="body"
437     xmlns:wsu="http://schemas.xmlsoap.org/ws/2003/06/utility">
438     <xenc:EncryptedData Id="enc"
439         Type="http://www.w3.org/2001/04/xmlenc#Content"
440         xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
441         <xenc:EncryptionMethod
442             Algorithm="http://www.w3.org/2001/04/xmlenc#tripleDES-cbc"/>
443         <xenc:CipherData>
444             <xenc:CipherValue>d2s...GQ=</xenc:CipherValue>
445         </xenc:CipherData>
446     </xenc:EncryptedData>
447 </S:Body>
448 </S:Envelope>

```

449 3.7 Error Codes

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

455

4 Types of Licenses (Informative)

456

4.1 Attribute Licenses

457

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

458

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

459

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

460

The following is an example client attribute license.

461

```
<r:license xmlns:r="..."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>
```

483

The following is an example service attribute license.

484

```
<r:license xmlns:r="..."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>
</r:license>
```

505

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

508 4.2 Sender Authorization

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

```
515 <r:license xmlns:r="..."licenseId="urn:foo:SecurityToken:ef375268">  
516   <r:grant>  
517     <r:keyHolder>  
518       <r:info>  
519         <ds:KeyValue>FDFEWEFF...</ds:KeyValue>  
520       </r:info>  
521     </r:keyHolder>  
522     <sx:rightUri definition='...'/>  
523     <x:someResource/>  
524     <x:someCondition/>  
525   </r:grant>  
526   <r:issuer>  
527     <ds:Signature>...</ds:Signature>  
528   </r:issuer>  
529 </r:license>
```

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

537 4.3 Issuer Authorization

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

543

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

```
546 <r:license xmlns:r="..."licenseId="urn:foo:SecurityToken:ef375268">  
547   <r:grant>  
548     <r:forAll varName='K' />  
549     <r:forAll varName='P' />  
550     <r:keyHolder>  
551       <r:info>  
552         <ds:KeyValue>FDFEWEFF...</ds:KeyValue>  
553       </r:info>
```

```

554     </r:keyHolder>
555     <r:issue/>
556     <r:grant>
557         <r:keyHolder varRef='K' />
558         <r:possessProperty/>
559         <r:propertyAbstract varRef='P' />
560     </r:grant>
561 </r:grant>
562 <r:issuer>
563     <ds:Signature>...</ds:Signature>
564 </r:issuer>
565 </r:license>

```

566 The following license is an example issuer authorization license for authorizing an issuer to issue
567 sender authorization licenses.

```

568 <r:license xmlns:r="..."licenseId="urn:foo:SecurityToken:ef375268">
569     <r:grant>
570         <r:forAll varName='K' />
571         <r:forAll varName='R' />
572         <r:keyHolder>
573             <r:info>
574                 <ds:KeyValue>FDFEWEFF...</ds:KeyValue>
575             </r:info>
576         </r:keyHolder>
577         <r:issue/>
578         <r:grant>
579             <r:keyHolder varRef='K' />
580             <sx:rightUri definition='...' />
581             <r:resource varRef='R' />
582         </r:grant>
583     </r:grant>
584     <r:issuer>
585         <ds:Signature>...</ds:Signature>
586     </r:issuer>
587 </r:license>

```

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

```

591 <r:license xmlns:r="..."licenseId="urn:foo:SecurityToken:ef375268">
592     <r:grant>
593         <r:forAll varName='I' />
594         <r:keyHolder>
595             <r:info>
596                 <ds:KeyValue>FDFEWEFF...</ds:KeyValue>
597             </r:info>
598         </r:keyHolder>
599         <r:issue/>
600         <r:grant>
601             <r:forAll varName='K' />
602             <r:forAll varName='P' />
603             <r:keyHolder varRef='I' />
604             <r:issue/>
605             <r:grant>
606                 <r:keyHolder varRef='K' />
607                 <r:possessProperty/>
608                 <r:propertyAbstract varRef='P' />
609             </r:grant>
610         </r:grant>
611     </r:grant>
612     <r:issuer>
613         <ds:Signature>...</ds:Signature>
614     </r:issuer>

```

615

</r:license>

616

617

5 Threat Model and Countermeasures (Informative)

618

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

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

636 The following few sections elaborate on the afore-mentioned threats and suggest
637 countermeasures.

638

5.1 Eavesdropping

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

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

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

650

5.2 Replay

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

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

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

657 **5.3 Message Insertion**

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

661 **5.4 Message Deletion**

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

665 **5.5 Message Modification**

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

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

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

678 **5.6 Man-in-the-Middle**

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

682

683

6 References

- 684 **[KEYWORDS]** S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels,"
685 RFC 2119, Harvard University, March 1997,
686 <http://www.ietf.org/rfc/rfc2119.txt>
- 687 **[REL]** ISO/IEC 21000-5:2004, "Information technology -- Multimedia framework
688 (MPEG-21) -- Part 5: Rights Expression Language,"
689 [http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUM
690 BER=36095&ICS1=35&ICS2=40&ICS3=](http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=36095&ICS1=35&ICS2=40&ICS3=)
- 691 **[SOAP]** D. Box, D Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H.
692 Frystyk Nielsen, S Thatte, D. Winer. Simple Object Access Protocol
693 (SOAP) 1.1, W3C Note 08 May 2000, <http://www.w3.org/TR/SOAP/>
- 694 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers
695 (URI): Generic Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox
696 Corporation, August 1998, <http://www.ietf.org/rfc/rfc2396.txt>
- 697 **[WS-Security]** OASIS Standard 200401, "Web Services Security: Soap Message
698 Security 1.0 (WS-Security 2004)," March 2004, [http://docs.oasis-
699 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)
- 700 **[XML-ns]** T. Bray, D. Hollander, A. Layman. Namespaces in XML. W3C
701 Recommendation. January 1999, [http://www.w3.org/TR/1999/REC-xml-
702 names-19990114](http://www.w3.org/TR/1999/REC-xml-names-19990114)
- 703 **[XML Signature]** D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. XML-
704 Signature Syntax and Processing, W3C Recommendation, 12 February
705 2002, <http://www.w3.org/TR/xmlsig-core/>
- 706

Appendix A: Revision History

Rev	Date	What
01	19-Sep-02	Initial draft produced by extracting SAML related content from [XML token]
02	12-Dec-02	Naming changes
03	30-Jan -03	Name changes, merged in comments from Thomas DeMartini
04	13-Nov-03	Updates, merged in comments from Thomas DeMartini
05	08-Jan-04	Contributor list updates, many title page updates, document name updates, namespace updates, switched from QNames to URIs.
06	29-Apr-04	Clarified case when a license contains more than one grant. Updated URIs.
07	24-May-04	Removed XrML from document name, per request from OASIS staff. Updated document to remove references to XrML.
08	18-Jun-04	Updated Contributor List

709 **Appendix B: Notices**

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

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

722 Copyright © OASIS Open 2002. *All Rights Reserved.*

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

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

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

739