



1

2

Web Services Security Rights Expression Language (REL) Token Profile 1.1

3

4

5

OASIS Standard: 1 February 2006

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-1.1.pdf>

10

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 Services Security (WSS) specification.

26

27

Status:

28

The status of this document is OASIS Standard. Please send comments to the editors.

WSS Rights Expression Language Token Profile

Copyright © OASIS Open 2002-2006. All Rights Reserved.

1 February 2006

Page 1 of 27

29 If you are on the wss@lists.oasis-open.org list for committee members, send comments
30 there. If you are not on that list, subscribe to the wss-comment@lists.oasis-open.org list
31 and send comments there. To subscribe, send an email message to [wss-comment-
request@lists.oasis-open.org](mailto:wss-comment-
32 request@lists.oasis-open.org) with the word "subscribe" as the body of the message.

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

Notices

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

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

51 Copyright © OASIS Open 2002-2006. All Rights Reserved.

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

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

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

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

Table of Contents

71	1	Introduction (Informative).....	5
72	2	Notations and Terminology (Normative).....	6
73	2.1	Notational Conventions.....	6
74	2.2	Namespaces.....	6
75	2.3	Terminology.....	7
76	3	Usage (Normative).....	8
77	3.1	Token Types.....	8
78	3.2	Processing Model.....	8
79	3.3	Attaching Security Tokens.....	8
80	3.4	Identifying and Referencing Security Tokens.....	8
81	3.5	Authentication.....	12
82	3.5.1	<r:keyHolder> Principal.....	12
83	3.6	Confidentiality.....	14
84	3.6.1	<r:keyHolder> Principal.....	15
85	3.7	Error Codes.....	16
86	4	Types of Licenses (Informative).....	17
87	4.1	Attribute Licenses.....	17
88	4.2	Sender Authorization.....	18
89	4.3	Issuer Authorization.....	18
90	5	Threat Model and Countermeasures (Informative).....	21
91	5.1	Eavesdropping.....	21
92	5.2	Replay.....	21
93	5.3	Message Insertion.....	22
94	5.4	Message Deletion.....	22
95	5.5	Message Modification.....	22
96	5.6	Man-in-the-Middle.....	22
97	6	References.....	23
98		Appendix A: Acknowledgements.....	24
99		Appendix B: Revision History.....	27

101

1 Introduction (Informative)

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

106 **2 Notations and Terminology (Normative)**

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

108 **2.1 Notational Conventions**

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

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

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

118 **2.2 Namespaces**

119 The following namespaces are used in this document:

120

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
----	-----------------------------------

121

Table 1 Namespace Prefixes

122

123 **2.3 Terminology**

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

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

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

128 3 Usage (Normative)

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

131 3.1 Token Types

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

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

137 3.2 Processing Model

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

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

143 3.3 Attaching Security Tokens

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

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

160 3.4 Identifying and Referencing Security Tokens

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

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

```

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

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

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

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

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

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

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

187 **Table 2. <wssu:SecurityTokenReference>**

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

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

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

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

```

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

```

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

```

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

```

293 **3.5 Authentication**

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

301 The following table enumerates the mandatory principals to be supported by claim confirmation
302 and summarizes their associated processing models. It should be noted that this table is not all-
303 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.

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

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

312 **3.5.1 <r:keyHolder> Principal**

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

315 **Sender**

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

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

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

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

329 Receiver

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

335 Example

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

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

372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391

```

    <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>

```

392 **3.6 Confidentiality**

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

400 The following table enumerates the mandatory principals to be supported for confidentiality and
401 summarizes their associated processing models. It should be noted that this table is not all-
402 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.

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

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

407 **3.6.1 <r:keyHolder> Principal**

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

410 **Sender**

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

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

422 **Receiver**

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

426

427 **Example**

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

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

436

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

```

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

```

485 3.7 Error Codes

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

491

4 Types of Licenses (Informative)

492

4.1 Attribute Licenses

493

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

494

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

495

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

496

The following is an example client attribute license.

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

```

<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>

```

520

The following is an example service attribute license.

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

```

<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 </r:license>

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

546 4.2 Sender Authorization

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

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

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

575 4.3 Issuer Authorization

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

581

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

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

```

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

```

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

```

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

```

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

```

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

```

651
652
653
654

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

655

5 Threat Model and Countermeasures (Informative)

656

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

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

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

676

5.1 Eavesdropping

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

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

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

688

5.2 Replay

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

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

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

695 **5.3 Message Insertion**

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

699 **5.4 Message Deletion**

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

703 **5.5 Message Modification**

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

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

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

716 **5.6 Man-in-the-Middle**

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

720

721

6 References

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

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

758 **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

759

760

Appendix B: Revision History

Rev	Date	What

761

762