Web Services Security Rights Expression Language (REL) Token Profile Version 1.1.1

Committee Specification Draft 01

23 March 2011

Specification URIs:
This Version:
> http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-rel-token-profile-v1.1.1-csd01.doc (Authoritative)
> http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-rel-token-profile-v1.1.1-csd01.pdf
> http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-rel-token-profile-v1.1.1-csd01.html

Previous Version:
> http://docs.oasis-open.org/wss/v1.1/oasis-wss-rel-token-profile-1.1.pdf

Latest Version:
> http://docs.oasis-open.org/wss-m/wss/v1.1.1/wss-rel-token-profile-v1.1.1.doc (Authoritative)
> http://docs.oasis-open.org/wss-m/wss/v1.1.1/wss-rel-token-profile-v1.1.1.pdf
> http://docs.oasis-open.org/wss-m/wss/v1.1.1/wss-rel-token-profile-v1.1.1.html

Technical Committee:
> OASIS Web Services Security Maintenance (WSS-M) TC

Chair(s):
> David Turner, Microsoft

Editor(s):
> Ronald Monzillo, Sun
> Chris Kaler, Microsoft
> Anthony Nadalin, IBM
> Phillip Hallam-Baker, VeriSign
> Thomas DeMartini, ContentGuard, Inc.
> Carlo Milono, Tibco

Related work:
> This specification is one part of a multi-part Work Product. The other parts include:

>> Web Services Security: SOAP Message Security Version 1.1.1
>> Web Services Security Kerberos Token Profile Version 1.1.1
>> Web Services Security SAML Token Profile Version 1.1.1
>> Web Services Security SOAP Messages with Attachments (SwA) Profile Version 1.1.1
>> Web Services Security Username Token Profile Version 1.1.1
>> Web Services Security X.509 Certificate Token Profile Version 1.1.1
>> Schemas: http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/xsd/

> This specification supersedes:

- Web Services Security Rights Expression Language (REL) Token Profile 1.1, OASIS Standard, 1 February 2006

Abstract:

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

Status:

This document was last revised or approved by the OASIS Web Services Security Maintenance (WSS-M) TC on the above date. The level of approval is also listed above. Check the "Latest Version" location noted above for possible later revisions of this document.

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

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (http://www.oasis-open.org/committees/wss-m/ipr.php).

This document integrates specific error corrections or editorial changes to the preceding specification, within the scope of the Web Services Security and this TC.

This document introduces a third digit in the numbering convention where the third digit represents a consolidation of error corrections, bug fixes or editorial formatting changes (e.g., 1.1.1); it does not add any new features beyond those of the base specifications (e.g., 1.1).

Citation Format:

[WSS-REL-Token-Profile-V-1.1.1]

Web Services Security Rights Expression Language (REL) Token Profile Version 1.1.1. 23 March 2011. OASIS Committee Specification Draft 01. http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-rel-token-profile-v1.1.1-csd01.doc.

Notices

Table of Contents

1 Introduction (Informative)

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

2 Notations and Terminology (Normative)

2.1 Notational Conventions

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

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

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

2.2 Namespaces

The following namespaces are used in this document:

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

20 *Table 1 Namespace Prefixes*

2.3 Terminology

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

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

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

26

3 Usage (Normative)

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

3.1 Token Types

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

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

3.2 Processing Model

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

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

3.3 Attaching Security Tokens

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

```
<S:Envelope xmlns:S="...">
    <S:Header>
        <wsse:Security xmlns:wsse="...">
            <r:license xmlns:r="...">
                ...
            </r:license>
            ...
        </wsse:Security>
    </S:Header>
    <S:Body>
        ...
    </S:Body>
</S:Envelope>
```

3.4 Identifying and Referencing Security Tokens

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

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

68 Licenses can be referenced either according to their location or their licenseId. Location references are
69 dependent on location and can be either local or remote. LicenseId references are not dependent on
70 location.

71 Local location references are RECOMMENDED when they can be used. Remote location references are
72 OPTIONAL for cases where it is not feasible to transmit licenses with the SOAP message. LicenseId
73 references are OPTIONAL for cases where location is unknown or cannot be indicated.

74 WS-Security specifies that tokens are referenced using the <wsse:SecurityTokenReference> element.
75 Implementations compliant with this profile SHOULD set the
76 /wsse:SecurityTokenReference/wsse:Reference/@ValueType attribute to http://docs.oasis-
77 open.org/wss/oasis-wss-rel-token-profile-1.0.pdf#license when using wsse:SecurityTokenReference to
78 refer to a license by licenseId. This is OPTIONAL when referring to a license by location.

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

| By Location | Local | ```<wsse:SecurityTokenReference>
  <wsse:Reference
    URI="#SecurityToken-ef375268"
  />
</wsse:SecurityTokenReference>``` | |
|---|---|---|---|
| | Remote | ```<wsse:SecurityTokenReference>
  <wsse:Reference
    URI="http://www.foo.com/ef375268.xml"
  />
</wsse:SecurityTokenReference>``` | |
| By licenseId | | ```<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>``` | |

81 *Table 2. <wsse:SecurityTokenReference>*

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

```
85  <S:Envelope xmlns:S="..." xmlns:ds="...">
86    <S:Header>
87      <wsse:Security xmlns:wsse="...">
88        <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268"
89  xmlns:wsu="..." wsu:Id="SecurityToken-ef375268">
90          ...
91        </r:license>
92        ...
93        <ds:Signature>
94          <ds:SignedInfo>
95            ...
96          </ds:SignedInfo>
97          <ds:SignatureValue>...</ds:SignatureValue>
98          <ds:KeyInfo>
99            <wsse:SecurityTokenReference>
100             <wsse:Reference
101               URI="#SecurityToken-ef375268"
102             />
103           </wsse:SecurityTokenReference>
```

```
104          </ds:KeyInfo>
105        </ds:Signature>
106      </wsse:Security>
107    </S:Header>
108    <S:Body>
109      ...
110    </S:Body>
111  </S:Envelope>
```

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

```
115  <S:Envelope xmlns:S="..." xmlns:wsu="..." >
116    <S:Header>
117      <wsse:Security xmlns:wsse="...">
118        <r:license xmlns:r="..." wsu:Id="SecurityToken-ef375268">
119          ...
120        </r:license>
121        ...
122        <wsse:SecurityTokenReference wsu:Id="Str1">
123          <wsse:Reference
124            URI="#SecurityToken-ef375268"
125          />
126        </wsse:SecurityTokenReference>
127        ...
128        <ds:Signature>
129          <ds:SignedInfo>
130            ...
131            <ds:Reference URI="#Str1">
132              <ds:Transforms>
133                <ds:Transform
134                  Algorithm="http://schemas.xmlsoap.org/2003/06/STR-Transform">
135                  <ds:CanonicalizationMethod
136                    Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
137  20010315"/>
138                </ds:Transform>
139              </ds:Transforms>
140              <ds:DigestMethod
141                Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
142              />
143              <ds:DigestValue>...</ds:DigestValue>
144            </ds:Reference>
145          </ds:SignedInfo>
146          <ds:SignatureValue>...</ds:SignatureValue>
147          <ds:KeyInfo>...</ds:KeyInfo>
148        </ds:Signature>
149      </wsse:Security>
150    </S:Header>
151    <S:Body>
152      ...
153    </S:Body>
154  </S:Envelope>
```

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

```
158  <S:Envelope xmlns:S="..." xmlns:ds="...">
159    <S:Header>
160      <wsse:Security xmlns:wsse="...">
```

```
161         <r:license xmlns:r="..." xmlns:wsu="..." wsu:Id="SecurityToken-
162    ef375268">
163           ...
164         </r:license>
165         ...
166         <ds:Signature>
167           <ds:SignedInfo>
168             ...
169             <ds:Reference URI="#SecurityToken-ef375268">
170               <ds:DigestMethod
171                 Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
172               />
173               <ds:DigestValue>...</ds:DigestValue>
174             </ds:Reference>
175           </ds:SignedInfo>
176           <ds:SignatureValue>...</ds:SignatureValue>
177           <ds:KeyInfo>...</ds:KeyInfo>
178         </ds:Signature>
179       </wsse:Security>
180     </S:Header>
181     <S:Body>
182       ...
183     </S:Body>
184  </S:Envelope>
```

185 3.5 Authentication

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

192 The following table enumerates the mandatory principals to be supported by claim confirmation and
193 summarizes their associated processing models. It should be noted that this table is not all-
194 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.

195 *Table 3. Processing Rules for Claim Confirmation*

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

3.5.1 <r:keyHolder> Principal

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

Sender

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

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

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

Receiver

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

Example

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

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

  <S:Header>
    <wsse:Security xmlns:wsse="...">

      <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
        <r:grant>
          <r:keyHolder>
            <r:info>
              <ds:KeyValue>...</ds:KeyValue>
            </r:info>
          </r:keyHolder>
          <r:possessProperty/>
          <sx:commonName xmlns:sx="...">John Doe</sx:commonName>
        </r:grant>
        <r:issuer>
          <ds:Signature>...</ds:Signature>
        </r:issuer>
      </r:license>

      <ds:Signature>
        <ds:SignedInfo>
          ...
          <ds:Reference URI="#MsgBody">
            <ds:DigestMethod
              Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"
```

```
253              />
254            <ds:DigestValue>...</ds:DigestValue>
255          </ds:Reference>
256        </ds:SignedInfo>
257        <ds:SignatureValue>...</ds:SignatureValue>
258        <ds:KeyInfo>
259          <wsse:SecurityTokenReference>
260            <wsse:Reference
261              URI="urn:foo:SecurityToken:ef375268"
262              ValueType="http://docs.oasis-open.org/wss/oasis-wss-rel-token-
263    profile-1.0.pdf#license"
264              />
265          </wsse:SecurityTokenReference>
266        </ds:KeyInfo>
267      </ds:Signature>
268
269    </wsse:Security>
270   </S:Header>
271
272   <S:Body wsu:Id="MsgBody" xmlns:wsu="...">
273     <ReportRequest>
274       <TickerSymbol>FOO</TickerSymbol>
275     </ReportRequest>
276   </S:Body>
277
278  </S:Envelope>
```

3.6 Confidentiality

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

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

Table 4. Processing Rules for Confidentiality

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

3.6.1 <r:keyHolder> Principal

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

Sender

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

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

Receiver

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

Example

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

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

319

```
320   <S:Envelope xmlns:S="..." xmlns:ds="...">
321     <S:Header>
322       <wsse:Security xmlns:wsse="...">
323         <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
324             <r:grant>
325               <r:keyHolder>
326                   <r:info>
327                     <ds:KeyValue>...</ds:KeyValue>
328                   </r:info>
329               </r:keyHolder>
330               <r:possessProperty/>
331               <sx:commonName xmlns:sx="...">SOME COMPANY</sx:commonName>
332             </r:grant>
333             <r:issuer>
334                   <ds:Signature>...</ds:Signature>
335             </r:issuer>
336         </r:license>
337         <xenc:EncryptedKey xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
338           <xenc:EncryptionMethod
339               Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
```

```
340              <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
341               <wsse:SecurityTokenReference>
342                    <wsse:Reference URI="urn:foo:SecurityToken:ef375268"/>
343               </wsse:SecurityTokenReference>
344              </KeyInfo>
345              <xenc:CipherData>
346                <xenc:CipherValue>dNYS...fQ=</xenc:CipherValue>
347              </xenc:CipherData>
348              <xenc:ReferenceList>
349                <xenc:DataReference URI="#enc"/>
350              </xenc:ReferenceList>
351            </xenc:EncryptedKey>
352         </wsse:Security>
353       </S:Header>
354       <S:Body wsu:Id="body"
355             xmlns:wsu="http://schemas.xmlsoap.org/ws/2003/06/utility">
356          <xenc:EncryptedData Id="enc"
357               Type="http://www.w3.org/2001/04/xmlenc#Content"
358               xmlns:xenc="http://www.w3.org/2001/04/xmlenc#">
359            <xenc:EncryptionMethod
360              Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
361            <xenc:CipherData>
362                <xenc:CipherValue>d2s...GQ=</xenc:CipherValue>
363            </xenc:CipherData>
364          </xenc:EncryptedData>
365       </S:Body>
366     </S:Envelope>
```

3.7 Error Codes

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

4 Types of Licenses (Informative)

4.1 Attribute Licenses

In addition to key information, licenses can carry information about attributes of those keys. Examples of such information on a client are e-mail address or common name. A service's key, on the other hand, might be associated with a DNS name and common name.

The following is an example client attribute license.

```
        <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>
```

The following is an example service attribute license.

```
        <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>
        </r:license>
```

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

4.2 Sender Authorization

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

```
<r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
    <r:grant>
      <r:keyHolder>
        <r:info>
          <ds:KeyValue>FDFEWEFF…</ds:KeyValue>
        </r:info>
     </r:keyHolder>
     <sx:rightUri definition='...'/>
     <x:someResource/>
     <x:someCondition/>
    </r:grant>
    <r:issuer>
       <ds:Signature>...</ds:Signature>
    </r:issuer>
</r:license>
```

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

4.3 Issuer Authorization

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

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

```
<r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
    <r:grant>
       <r:forAll varName='K'/>
       <r:forAll varName='P'/>
      <r:keyHolder>
         <r:info>
           <ds:KeyValue>FDFEWEFF…</ds:KeyValue>
         </r:info>
      </r:keyHolder>
      <r:issue/>
      <r:grant>
        <r:keyHolder varRef='K'/>
        <r:possessProperty/>
        <r:propertyAbstract varRef='P'/>
      </r:grant>
     </r:grant>
     <r:issuer>
        <ds:Signature>...</ds:Signature>
     </r:issuer>
</r:license>
```

484 The following license is an example issuer authorization license for authorizing an issuer to issue sender
485 authorization licenses.

```
486        <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
487            <r:grant>
488              <r:forAll varName='K'/>
489              <r:forAll varName='R'/>
490              <r:keyHolder>
491                <r:info>
492                  <ds:KeyValue>FDFEWEFF…</ds:KeyValue>
493                </r:info>
494              </r:keyHolder>
495              <r:issue/>
496              <r:grant>
497                <r:keyHolder varRef='K'/>
498                <sx:rightUri definition='...'/>
499                <r:resource varRef='R'/>
500              </r:grant>
501            </r:grant>
502            <r:issuer>
503              <ds:Signature>...</ds:Signature>
504            </r:issuer>
505        </r:license>
```

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

```
509        <r:license xmlns:r="..." licenseId="urn:foo:SecurityToken:ef375268">
510          <r:grant>
511            <r:forAll varName='I'/>
512            <r:keyHolder>
513              <r:info>
514                <ds:KeyValue>FDFEWEFF…</ds:KeyValue>
515              </r:info>
516            </r:keyHolder>
517            <r:issue/>
518            <r:grant>
519              <r:forAll varName='K'/>
520              <r:forAll varName='P'/>
521              <r:keyHolder varRef='I'/>
522              <r:issue/>
523              <r:grant>
524                <r:keyHolder varRef='K'/>
525                <r:possessProperty/>
526                <r:propertyAbstract varRef='P'/>
527              </r:grant>
528            </r:grant>
529          </r:grant>
530          <r:issuer>
531            <ds:Signature>...</ds:Signature>
532          </r:issuer>
533        </r:license>
```

534

5 Threat Model and Countermeasures (Informative)

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

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

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

5.1 Eavesdropping

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

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

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

5.2 Replay

The reliance on authority protected (e.g. signed) licenses to <r:keyHolder> principals precludes all but the key holder from binding the licenses to a SOAP message. Although this mechanism effectively restricts message authorship to the holder of the confirmation key, it does not preclude the capture and resubmission of the message by other parties.

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

5.3 Message Insertion

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

5.4 Message Deletion

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

5.5 Message Modification

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

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

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

5.6 Man-in-the-Middle

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

6 References

[KEYWORDS]   S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," RFC 2119, Harvard University, March 1997, http://www.ietf.org/rfc/rfc2119.txt

[REL]   ISO/IEC 21000-5:2004, "Information technology -- Multimedia framework (MPEG-21) -- Part 5: Rights Expression Language," http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=36095&ICS1=35&ICS2=40&ICS3=

[SOAP]   D. Box, D Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. Frystyk Nielsen, S Thatte, D. Winer. Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000, http://www.w3.org/TR/SOAP/

W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging Framework", 23 June 2003

[URI]   T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998, http://www.ietf.org/rfc/rfc2396.txt

T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 3986, MIT/LCS, Day Software, Adobe Systems, January 2005, http://www.ietf.org/rfc/rfc3986.txt.

[WS-Security]   Nadalin et al., "Web Services Security: SOAP Message Security 1.1.1 http://docs.oasis-open.org/wss-m/wss/v1.1.1/csd01/wss-SOAPMessageSecurity-v1.1.1-csd01.pdf

OASIS Standard, "Web Services Security: Soap Message Security 1.1 (WS-Security 2004)," November 2005, http://docs.oasis-open.org/wss/ oasis-wss-soap-message-security-1.1.pdf

[XML-ns]   T. Bray, D. Hollander, A. Layman. Namespaces in XML. W3C Recommendation. January 1999, http://www.w3.org/TR/1999/REC-xml-names-19990114

[XML Signature]   D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. XML-Signature Syntax and Processing, W3C Recommendation, 12 February 2002.

7 Conformance

628 The last numbered section in the specification must be the Conformance section. Conformance
629 Statements/Clauses go here.

A. Acknowledgements

630

631 The following individuals have participated in the creation of this specification and are gratefully
632 acknowledged:

633 **Participants:**
634 **Current Contributors:**

Tom	Rutt	Fujitsu Limited
Jacques	Durand	Fujitsu Limited
Calvin	Powers	IBM
Kelvin	Lawrence	IBM
Michael	McIntosh	Individual
Thomas	Hardjono	M.I.T.
David	Turner	Microsoft Corporation
Anthony	Nadalin	Microsoft Corporation
Monica	Martin	Microsoft Corporation
Marc	Goodner	Microsoft Corporation
Peter	Davis	Neustar
Hal	Lockhart	Oracle Corporation
Rich	Levinson	Oracle Corporation
Anil	Saldhana	Red Hat
Martin	Raepple	SAP AG
Federico	Rossini	Telecom Italia S.p.a.
Carlo	Milono	TIBCO Software Inc.
Don	Adams	TIBCO Software Inc.
Jerry	Smith	US Department of Defense (DoD)

635 **Previous Contributors:**

Michael	Hu	Actional
Maneesh	Sahu	Actional
Duane	Nickull	Adobe Systems
Gene	Thurston	AmberPoint
Frank	Siebenlist	Argonne National Laboratory
Peter	Dapkus	BEA
Hal	Lockhart	BEA Systems
Denis	Pilipchuk	BEA Systems
Corinna	Witt	BEA Systems
Steve	Anderson	BMC Software
Rich	Levinson	Computer Associates
Thomas	DeMartini	ContentGuard
Guillermo	Lao	ContentGuard
TJ	Pannu	ContentGuard
Xin	Wang	ContentGuard

Merlin	Hughes	Cybertrust
Dale	Moberg	Cyclone Commerce
Shawn	Sharp	Cyclone Commerce
Rich	Salz	Datapower
Ganesh	Vaideeswaran	Documentum
Sam	Wei	EMC
John	Hughes	Entegrity
Tim	Moses	Entrust
Carolina	Canales-Valenzuela	Ericsson
Davanum	Srinivas	formerly of Computer Associates
Mark	Hayes	formerly of VeriSign
Dana S.	Kaufman	Forum Systems
Toshihiro	Nishimura	Fujitsu
Tom	Rutt	Fujitsu
Kefeng	Chen	GeoTrust
Irving	Reid	Hewlett-Packard
Kojiro	Nakayama	Hitachi
Yutaka	Kudo	Hitachi
Jason	Rouault	HP
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
Bob	Blakley	IBM
Joel	Farrell	IBM
Satoshi	Hada	IBM
Hiroshi	Maruyama	IBM
David	Melgar	IBM
Kent	Tamura	IBM
Wayne	Vicknair	IBM
Don	Flinn	Individual
Phil	Griffin	Individual
Bob	Morgan	Individual/Internet2
Kate	Cherry	Lockheed Martin
Paul	Cotton	Microsoft
Vijay	Gajjala	Microsoft

Martin	Gudgin	Microsoft
Chris	Kaler	Microsoft
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
Frederick	Hirsch	Nokia
Senthil	Sengodan	Nokia
Abbie	Barbir	Nortel
Lloyd	Burch	Novell
Ed	Reed	Novell
Charles	Knouse	Oblix
Vamsi	Motukuru	Oracle
Vipin	Samar	Oracle
Jerry	Schwarz	Oracle
Prateek	Mishra	Principal Identity
Eric	Gravengaard	Reactivity
Stuart	King	Reed Elsevier
Ben	Hammond	RSA Security
Rob	Philpott	RSA Security
Andrew	Nash	RSA Security
Peter	Rostin	RSA Security
Martijn	de Boer	SAP
Blake	Dournaee	Sarvega
Sundeep	Peechu	Sarvega
Pete	Wenzel	SeeBeyond
Jonathan	Tourzan	Sony
Yassir	Elley	Sun
Manveen	Kaur	Sun Microsystems
Ronald	Monzillo	Sun Microsystems

Jan	Alexander	Systinet
Michael	Nguyen	The IDA of Singapore
Don	Adams	TIBCO
Symon	Chang	TIBCO Software
John	Weiland	US Navy
Hans	Granqvist	VeriSign
Phillip	Hallam-Baker	VeriSign
Hemma	Prafullchandra	VeriSign
Morten	Jorgensen	Vordel

636
637

638 # B. Revision History

639

Revision	Date	Editor	Changes Made
WD01	17-January-2011	Carlo Milono	Corrected/added hyperlinks where missing; added Status section
WD02	8-February-2011	Carlo Milono	Added Related Work to reflect v1.1.1 of the specs; changed References for SOAP Message Security to reflect v1.1.1; Changed WD# to 2; Added Date; Moved Current Members to Previous and added new Current Members; saved document under wd02; entered the Revision History

Merged Old Current Contributors with Old Previous, created a New Current Contributors. |

640
641