



1 Web Services Security 2 SOAP Messages with Attachments 3 (SwA) Profile 1.0

4 OASIS Committee Draft 01, 23 December 2004

5 **Document identifier:**

6 wss-swa-profile-1.0-cd-01

7 **Location:**

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

9 **Editors:**

10 Frederick Hirsch, Nokia

11 **Contributors:**

12 Michael Hu, Actional
13 Maneesh Sahu, Actional
14 Thomas DeMartini, ContentGuard
15 Dale Moberg, Cyclone Commerce
16 Dana S. Kaufman, Forum Systems, Inc
17 Dan Smiley, Forum Systems, Inc
18 Michael McIntosh, IBM
19 Frederick Hirsch, Nokia
20 Jerry Schwarz, Oracle
21 Blake Dournaee, Sarvega, Inc.
22 Pete Wenzel, SeeBeyond

23 **Abstract:**

24 This specification defines how to use the OASIS Web Services Security: SOAP Message Security
25 standard [WSS-Sec] with SOAP Messages with Attachments [SwA].

26 **Status:**

27 This is a Committee Draft.

28 Committee members should submit comments and potential errata to the [wss@lists.oasis-](mailto:wss@lists.oasis-open.org)
29 [open.org](mailto:wss@lists.oasis-open.org) list. Others should submit them to the wss-comment@lists.oasis-open.org list (to post,
30 you must subscribe; to subscribe, send a message to [wss-comment-subscribe@lists.oasis-](mailto:wss-comment-subscribe@lists.oasis-open.org)
31 [open.org](mailto:wss-comment-subscribe@lists.oasis-open.org) with "subscribe" in the body) or use other OASIS-supported means of submitting
32 comments.

33 For information on whether any patents have been disclosed that may be essential to
34 implementing this specification, and any offers of patent licensing terms, please refer to the
35 Intellectual Property Rights web page for the WSS TC ([http://www.oasis-](http://www.oasis-open.org/committees/wss/ipr.php)
36 [open.org/committees/wss/ipr.php](http://www.oasis-open.org/committees/wss/ipr.php)).

37 **Table of Contents**

38	1 Introduction.....	3
39	1.1 Notations and Terminology.....	3
40	1.1.1 Notational Conventions.....	3
41	1.1.2 Namespaces.....	4
42	1.1.3 Acronyms and Abbreviations.....	4
43	2 MIME Processing.....	5
44	3 XML Attachments.....	6
45	4 Securing SOAP With Attachments.....	7
46	4.1 Primary SOAP Envelope.....	7
47	4.2 Referencing Attachments.....	7
48	4.3 MIME Part Reference Transforms.....	7
49	4.3.1 Attachment-Content-Only Reference Transform.....	8
50	4.3.2 Attachment-Complete Reference Transform.....	8
51	4.4 Integrity and Data Origin Authentication	8
52	4.4.1 MIME header canonicalization.....	9
53	4.4.2 MIME Content Canonicalization.....	10
54	4.4.3 Protecting against attachment insertion threat.....	10
55	4.4.4 Processing Rules for Attachment Signing.....	11
56	4.4.5 Processing Rules for Attachment Signature Verification.....	11
57	4.4.6 Example Signed Message.....	12
58	4.5 Encryption.....	12
59	4.5.1 MIME Part CipherReference.....	13
60	4.5.2 Encryption Processing Rules.....	13
61	4.5.3 Decryption Processing Rules.....	14
62	4.5.4 Example.....	15
63	4.6 Signing and Encryption.....	16
64	5 References.....	17

1 Introduction

This document describes how to use the OASIS Web Services Security: SOAP Message Security standard [WSS-Sec] with SOAP Messages with Attachments [SwA]. More specifically, it describes how a web service consumer can secure SOAP attachments using SOAP Message Security for attachment integrity, confidentiality and origin authentication, and how a receiver may process such a message.

A broad range of industries - automotive, insurance, financial, pharmaceutical, medical, retail, etc - require that their application data be secured from its originator to its ultimate consumer. While some of this data will be XML, quite a lot of it will not be. In order for these industries to deploy web service solutions, they need an interoperable standard for end-to-end security for both their XML data and their non-XML data.

Profiling SwA security may help interoperability between the firms and trading partners using attachments to convey non-XML data that is not necessarily linked to the XML payload. Many industries, such as the insurance industry require free-format document exchange in conjunction with web services messages. This profile of SwA should be of value in these cases.

In addition, some content that could be conveyed as part of the SOAP body may be conveyed as an attachment due to its large size to reduce the impact on message and XML processing, and may be secured as described in this profile.

This profile is applicable to using SOAP Message Security in conjunction with SOAP Messages with Attachments (SwA). This means the scope is limited to SOAP 1.1, the scope of SwA.

The existence of this profile does not preclude using other mechanisms to secure attachments conveyed in conjunction with SOAP messages, including the use of XML security technologies at the application layer or the use of security for the XML Infoset before a serialization that uses attachment technology [MTOM]. The requirements in this profile only apply when securing SwA attachments explicitly according to this profile.

Note that in this document, lists of processing steps are descriptive in that an implementation may use a different procedure as long as the result is the same.

1.1 Notations and Terminology

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

1.1.1 Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC 2119 [RFC2119].

Listings of productions or other normative code appear like this.

Example code listings appear like this.

Note: Non-normative notes and explanations appear like this.

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

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

106 Commonly used security terms are defined in the Internet Security Glossary [SECGLO]. Readers are
107 presumed to be familiar with the terms in this glossary as well as the definitions in the SOAP Message
108 Security specification [WSS-Sec] .

109 1.1.2 Namespaces

110 Namespace URIs (of the general form "some-URI") represent application-dependent or context-
111 dependent URIs as defined in RFC 2396 [URI]. This specification is designed to work with the SOAP 1.1
112 [SOAP11] message structure and message processing model, the version of SOAP supported by SOAP
113 Messages with Attachments. The current SOAP 1.1 namespace URI is used herein to provide detailed
114 examples.

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

Prefix	Namespace
S11	http://schemas.xmlsoap.org/soap/envelope/
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
wsswa	http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0.xsd

117 The URLs provided for the *wsse* and *wsu* namespaces can be used to obtain the schema files.

118 **Note: When this document is finalized the wsswa URL will be updated, replacing**
119 **XX values and possibly making other changes.**

120 1.1.3 Acronyms and Abbreviations

121 The following (non-normative) table defines acronyms and abbreviations for this document, beyond those
122 defined in the SOAP Message Security standard.

Term	Definition
CID	Content ID scheme for URLs. Refers to Multipart MIME body part, that includes both MIME headers and content for that part. [RFC2392]
SwA	SOAP Messages with Attachments [SwA]

2 MIME Processing

124 This profile is concerned with the securing SOAP messages with attachments, attachments that are
125 conveyed as MIME parts in a multi-part MIME message as outlined in SOAP Messages with Attachments.
126 In effect this combines two processing layers, a SOAP messaging layer and a MIME wrapping. A SOAP
127 sender effectively transmits a SOAP message and corresponding attachments by passing them to a
128 MIME layer that serializes them. A SOAP receiver receives a message and attachments after the MIME
129 layer processes the MIME serialization. This is important since certain aspects of the MIME processing
130 may be changed at different intermediary transport nodes, yet remain transparent to the SOAP layer. For
131 example, a MIME processing node may change the transfer encoding of a MIME part, transparently to the
132 SOAP nodes. The MIME layer may translate to and from a transfer encoding upon serialization and de-
133 serialization.

134 The importance to this profile is two-fold. First, it means that certain aspects of MIME processing, such as
135 transfer encoding processing, are out of scope of the profile and do not need to be addressed. Secondly, it
136 means that many of the MIME headers are also out of scope of the profile and the profile does not support
137 integrity protection of these headers, since they are expected to change. If more security protection is
138 required then it must occur at a protocol layer below the MIME layer, for example transport security (with
139 the understanding that such security may not always apply end-end).

140 SOAP message security is intended to provide security at the SOAP messaging layer, including support
141 for SOAP intermediaries. Thus this profile supports securing the attachment content, possibly including
142 MIME headers that are associated directly with the content (such as Content-Type, Content-Length and
143 other Content related MIME headers) and not MIME headers associated with MIME serialization. This
144 simplifies the profile and also delineates the layering.

145

3 XML Attachments

146 A SOAP Messages with Attachments multi-part MIME structure contains a primary SOAP envelope in the
147 root part and one or more attachments in additional MIME parts. Some of these attachments may be have
148 a content type corresponding to XML, but do not contain the primary SOAP envelope to be processed.

149 Attachments associated with the SOAP body are targeted at the SOAP Ultimate Receiver along with the
150 SOAP body, and may be processed at the application layer along with the body. This means that XML
151 processing may not be required for such XML media type MIME attachments until application layer
152 processing is performed. For this reason the SOAP message layer may not need to perform XML
153 canonicalization or parsing for such attachments and SOAP Message layer security may treat these
154 attachments as text.

155 Attachments might also be associated with SOAP headers and targeted toward specific SOAP
156 intermediaries, or actors. For SOAP headers specific to an application the attachment content is
157 processed at the application layer, logically after SOAP message processing is complete.

158 This profile assumes that SOAP attachments (not including the root part containing the primary SOAP
159 envelope) need not be processed as XML at the SOAP messaging layer, so do not require SOAP
160 canonicalization or XML parsing and may be treated as opaque data by the SOAP Message Security layer
161 security processing. MIME part canonicalization (as described below) is required to enable effective SOAP
162 Message Security signatures that include SOAP with Attachments.

163 4 Securing SOAP With Attachments

164 Attachments may be associated with SOAP messages, as outlined in SOAP Messages with Attachments
165 (SwA). This profile defines how such attachments may be secured for integrity and confidentiality using
166 the OASIS Web Services Security: SOAP Message Security standard. This does not preclude using other
167 techniques. The requirements in this profile only apply when securing SwA attachments explicitly
168 according to this profile.

169 This profile considers all attachments as opaque whether they are XML or some other content type. It is
170 the sole responsibility of the application to perform further interpretation of attachments that happen to be
171 XML, including the ability to sign or encrypt portions of those attachments.

172 4.1 Primary SOAP Envelope

173 When SOAP attachments are used as specified in [SwA] each SOAP message is accompanied by a
174 MIME header and possibly multiple boundary parts. This is known as a SOAP message package. This
175 document assumes that a proper SOAP message package is constructed using the HTTP and MIME
176 headers appropriate to [SwA].

177 The primary SOAP envelope SHOULD be conveyed in the first MIME part, but MAY be conveyed in
178 another MIME part when the start attribute is specified in the HTTP Multipart/Related header.

179 In particular, implementations should take care in distinguishing between the HTTP headers in the SOAP
180 message package and the start of the SOAP payload. For example, the following Multipart/Related
181 header belongs to the HTTP layer and not the main SOAP payload:

```
182 Content-Type: Multipart/Related; boundary=xyl; type="text/xml"; start="<foo>"
```

183 The main SOAP payload begins with the appropriate boundary. For example:

```
184 --xyl  
185 Content-Type: text/xml; charset=utf-8  
186 Content-ID: <foo>  
  
187 <?xml version='1.0' ?>  
188 <s11:Envelope xmlns:s11="http://schemas.xmlsoap.org/soap/envelope/" />
```

189 4.2 Referencing Attachments

190 SOAP Messages with Attachments defines two MIME mechanisms for referencing attachments. The first
191 mechanism uses a CID scheme URL to refer to the attachment that has a Content-ID MIME header with a
192 value corresponding to the URL, as defined in [RFC 2392]. For example, a content id of "foo" may be
193 specified in the MIME part with the MIME header "Content-ID: <foo>" and be referenced using the CID
194 Schema URL "cid:foo".

195 The second mechanism is to use a URL to refer to an attachment containing a Content-Location MIME
196 header. In this case the URL may require resolution to determine the referenced attachment [RFC2557].

197 For simplicity and interoperability this profile limits WS-Security references to attachments to CID scheme
198 URLs. Attachments referenced from WS-Security signature references or cipher references MUST be
199 referenced using CID scheme URLs.

200 4.3 MIME Part Reference Transforms

201 By definition of RFC 2392, a URI reference to a MIME attachment includes the MIME headers associated

202 with that attachment as well as the MIME part content [RFC2392]. Since there may be some confusion as
203 to what is referenced, it is useful to clearly indicate what is included in the referenced attachment. In
204 addition, some applications may wish to only encrypt or include the attachment content in a signature
205 reference hash, and others may wish to include MIME headers and content.

206 For these reasons, this profile defines two transforms, allowing a clear and explicit statement of what is
207 included in a MIME reference. These transforms are called "MIME Part Reference Transforms".

208 **4.3.1 Attachment-Content-Only Reference Transform**

209 The Attachment-Content-Only transform indicates that only the content of a MIME part is referenced. This
210 transform MUST be identified using the URI value: [http://docs.oasis-open.org/wss/2004/XX/oasis-
211 2004XX-wss-swa-profile-1.0#Attachment-Content-Only-Transform](http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-Only-Transform).

212 **Note: When this document is finalized this URL will be updated, replacing XX**
213 **values and possibly making other changes.**

214 When this transform is used the content of the MIME part should be MIME canonicalized as defined in
215 section 4.4.2.

216 **4.3.2 Attachment-Complete Reference Transform**

217 The Attachment-Complete transform indicates that both the content and selected headers of the MIME
218 part are referenced. This transform MUST be identified using the URI value: [http://docs.oasis-
219 open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Complete-Transform](http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Complete-Transform).

220 **Note: When this document is finalized this URL will be updated, replacing XX**
221 **values and possibly making other changes.**

222 This transform specifies that in addition to the content the following MIME headers are to be included
223 (when present):

- 224 • Content-Description
- 225 • Content-Disposition
- 226 • Content-ID
- 227 • Content-Location
- 228 • Content-Type

229 These headers are included because of their common use and the risks associated with inappropriate
230 modification. If other headers are to be protected, other mechanisms at the application level should be
231 used (such as copying values into a SOAP header) and this is out of scope of this profile.

232 Other MIME headers associated with the MIME part serialization are not referenced by the transform and
233 are not to be included in signature or encryption calculations.

234 When this transform is used the MIME headers should be canonicalized as defined in section 4.4.1 and
235 the MIME content should be MIME canonicalized as defined in section 4.4.2.

236 **4.4 Integrity and Data Origin Authentication**

237 Integrity and data origin authentication may be provided for SwA attachments using XML Digital
238 Signatures, as outlined in the SOAP Message Security standard as profiled in this document. This is
239 useful independent of the content of the MIME part – for example, it is possible to sign a MIME part that
240 already contains a signed object created by an application. It may still be sensible to sign such an
241 attachment as part of SOAP Message security so that the receiving SOAP node may verify that all
242 attachments are intact before delivering them to an application. A SOAP intermediary may also choose to
243 perform this verification, even if the attachments are not otherwise processed by the intermediary.

244 4.4.1 MIME header canonicalization

245 The result of MIME header canonicalization is a UTF-8 encoded octet stream.

246 Each of the MIME headers listed for the Attachment-Complete transform MUST be canonicalized as part
247 of that transform processing, as outlined in this section. This means the transform MUST perform the
248 following actions in interpreting the MIME headers for signature creation or verification (this order is not
249 prescriptive as long as the same result is obtained)

- 250 1. The transform MUST process MIME headers before the MIME content.
- 251 2. The transform MUST only process MIME headers that are explicitly present in the attachment part and
252 are listed in the Attachment-Complete transform section of this specification, except that a MIME part
253 without a Content-Type header MUST be treated as having a Content-Type header with the value
254 "Content-Type: text/plain; charset=us-ascii". MIME headers not listed in the Attachment-Complete
255 transform section of this specification are to be ignored by the transform.
- 256 3. The MIME headers MUST be processed by the Attachment-Complete transform in lexicographic order
257 (ascending).
- 258 4. The MIME header names MUST be processed by the transform as having the case according to the
259 MIME specifications (as shown in the Attachment-Complete section).
- 260 5. The MIME header values MUST be unfolded [RFC2822].
- 261 6. Any Content-Description MIME header containing RFC2047 encoding MUST be decoded [RFC2047].
- 262 7. When a Content-ID header is processed, the "<>" characters associated with the msg-id MUST be
263 included in the transform input. The reason is that although semantically these angle bracket
264 characters are not part of the msg-id (RFC 2822) they are a standard part of the header lexicographic
265 representation. If these characters are not integrity protected then an attacker could remove them
266 causing the CID transformation specified in RFC2392 to fail.
- 267 8. Folding whitespace in structured MIME headers (e.g. Content-Disposition, Content-ID, Content-
268 Location, Content-Type) that is not within quotes MUST be removed. Folding whitespace in structured
269 MIME headers that is within quotes MUST be preserved. Folding whitespace in unstructured MIME
270 headers (e.g. Content-Description) MUST be preserved [RFC2822]. For example, whitespace
271 immediately following the colon delimiter in the structured Content-Type header MUST be removed,
272 but whitespace immediately following the colon delimiter in the unstructured Content-Description
273 header MUST be preserved.
- 274 9. Comments in MIME header values MUST be removed [RFC2822].
- 275 10. Case-insensitive MIME header values (e.g. media type/subtype values and disposition-type values)
276 MUST be converted to lowercase. Case-sensitive MIME header values MUST be left as is with
277 respect to case [RFC2045].
- 278 11. Quoted characters other than double-quote and backslash ("\") in quoted strings in structured MIME
279 headers (e.g. Content-ID) MUST be unquoted. Double-quote and backslash ("\") characters in quoted
280 strings in structured MIME headers MUST be character encoded [RFC2822].
- 281 12. Canonicalization of a MIME header MUST generate a UTF-8 encoded octet stream containing the
282 following: the MIME header name, a colon (":"), the MIME header value, and the result of
283 canonicalizing the MIME header parameters in lexicographic order (ascending) as described below.
- 284 13. MIME header parameter names MUST be converted to lowercase [RFC2045].
- 285 14. MIME parameter values containing RFC2184 character set, language, and continuations MUST be
286 decoded. The resulting canonical output MUST not contain the RFC2184 encoding [RFC2184].
- 287 15. Case-insensitive MIME header parameter values MUST be converted to lowercase. Case-sensitive
288 MIME header parameter values MUST be left as is with respect to case [RFC2045].

- 289 16. Enclosing double-quotes MUST be added to MIME header parameter values that do not already
290 contain enclosing quotes. Quoted characters other than double-quote and backslash ("\") in MIME
291 header parameter values MUST be unquoted. Double-quote and backslash characters in MIME
292 parameter values MUST be character encoded.
- 293 17. Canonicalization of a MIME header parameter MUST generate a UTF-8 encoded octet stream
294 containing the following: a semi-colon (";"), the parameter name (lowercase), an equals sign ("="), and
295 the double-quoted parameter value.
- 296 18. Each header MUST be terminated by a single CRLF pair, without any trailing whitespace.
- 297 19. The last header MUST be followed by a single CRLF and then the MIME content.

298 **4.4.2 MIME Content Canonicalization**

299 Before including attachment content in a signature reference hash calculation, that MIME attachment may
300 need to be MIME canonicalized. The exact details of MIME part canonicalization depend on the Content-
301 Type of the MIME part. To quote the S/MIME specification (section 3.1.1 "Canonicalization") which deals
302 with this issue [[RFC2633](#)]:

303 The exact details of canonicalization depend on the actual MIME type and subtype of an
304 entity, and are not described here. Instead, the standard for the particular MIME type should
305 be consulted. For example, canonicalization of type text/plain is different from
306 canonicalization of audio/basic. Other than text types, most types have only one
307 representation regardless of computing platform or environment which can be considered
308 their canonical representation.

309 MIME types are registered. This registration includes a section on "Canonicalization and Format
310 Requirements" [[RFC2048](#)] and requires each MIME type to have a canonical representation.

311 The MIME "text" type canonical form is defined in the MIME conformance specification (See "Canonical
312 Encoding Model") [[RFC2049](#)]. Important aspects of "text" media type canonicalization include line ending
313 normalization to <CR><LF> and ensuring that the charset is a registered charset (see RFC 2633 section
314 "Canonicalization"). [[RFC2633](#), [CHARSETS](#), [RFC2045](#)].

315 MIME attachment parts (other than the part containing the primary SOAP envelope) that contain XML do
316 NOT require XML Canonicalization according to this profile, given the rationale in the previous section on
317 XML attachments. These parts MUST be MIME canonicalized according to the MIME "text" part
318 requirements. MIME part canonicalization must be performed before signature hash generation or
319 verification is performed. Signature validation requires an identical hash of content requiring consistent
320 MIME part content.

321 **4.4.3 Protecting against attachment insertion threat**

322 Including an attachment in a signature calculation enables a receiver to detect modification of that
323 attachment. Including all attachments in a signature calculation, by providing a <ds:Reference> for each,
324 protects against the threat of attachment removal. This does not protect against insertion of a new
325 attachment.

326 The simplest protection against attachment insertion is for the receiver to know that all attachments
327 should be included in a signature calculation – unreferenced attachments are then an indication of an
328 attachment insertion attack.

329 Such information may be communicated in or out of band. Definition of these approaches is out of the
330 scope of this profile.

331 4.4.4 Processing Rules for Attachment Signing

332 The processing rule for signing is modified based on the SOAP Message Security rules.

333 After determining which attachments are to be included as references in a signature, create a
334 <ds:Signature> element in a <wsse:Security> header block targeted at the recipient, including a
335 <ds:Reference> for each attachment to be protected by the signature. Additional <ds:Reference>
336 elements may refer to content in the SOAP envelope to be included in the signature.

337 For each attachment Reference, perform the following steps:

- 338 1. MIME Part Canonicalize the content of the attachment, as appropriate to the MIME type of the part.
- 339 2. If MIME headers are to be included in the signature, MIME part canonicalize the headers listed in this
340 profile as outlined above.
- 341 3. Determine the CID scheme URL to be used to reference the part and set the <ds:Reference> URL
342 attribute value to this URL.
- 343 4. Include a <ds:Transforms> element in the <ds:Reference>. This <ds:Transforms> element MUST
344 include a <ds:Transform> element with the Algorithm attribute having the URL value specified in this
345 profile - either Attachment-Complete or Attachment-Content-Only, depending on what is to be
346 included in the hash calculation. This MUST be the first transform listed. The <ds:Transform> element
347 MUST NOT contain any transform for a transfer encoding purpose (e.g. base64 encoding) since
348 transfer encoding is left to the MIME layer as noted in section 2.
- 349 5. Extract the appropriate portion of the MIME part consistent with the selected transform.
- 350 6. Create the <ds:Reference> hash value as outlined in the W3C XML Digital Signature
351 Recommendation.

352 4.4.5 Processing Rules for Attachment Signature Verification

353 Signature verification is performed as outlined in SOAP Message Security and the XML Digital Signature
354 Recommendation, with the following considerations for SwA attachments.

355 To verify <ds:Reference> hashes for SwA attachments, the following steps must be performed for each
356 reference to an attachment:

- 357 1. Find the attachment corresponding to the <ds:Reference> URL attribute value. This value MUST
358 correspond to the Content-ID for the attachment[SwA].
- 359 2. MIME Part Canonicalize the content of the attachment, as appropriate to the MIME type of the part.
360 The MIME content to be MIME canonicalized MUST have had any transfer-encoding processed at the
361 MIME layer before this step is performed.
- 362 3. If MIME headers were included in the signature, canonicalize the headers listed in this profile as
363 outlined above.
- 364 4. Extract the appropriate portion of the MIME part according to the MIME Part Signature Transform
365 value.
- 366 5. Calculate the reference hash and verify the reference.

367 4.4.6 Example Signed Message

```
368 Content-Type: multipart/related; boundary="BoundaryStr" type="text/xml"
369 --BoundaryStr
370 Content-Type: text/xml
371 <S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..."
372 xmlns:ds="..." xmlns:xenc="...">
373   <S11:Header>
374     <wsse:Security>
375       <wsse:BinarySecurityToken wsu:Id="CertAssociatedWithSigningKey"
376         EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
377 200401-wss-soap-message-security-1.0#Base64Binary"
378         ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
379 200401-wss-x509-token-profile-1.0#x509v3">
380         ...
381       </wsse:BinarySecurityToken>
382     <ds:Signature>
383       <ds:SignedInfo>
384         <ds:CanonicalizationMethod Algorithm=
385 'http://www.w3.org/2001/10/xml-exc-c14n#'/>
386         <ds:SignatureMethod Algorithm=
387 'http://www.w3.org/2000/09/xmldsig#rsa-sha1' />
388         <ds:Reference URI="cid:bar">
389           <ds:Transforms>
390             <ds:Transform Algorithm="http://docs.oasis-
391 open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-
392 Only-Transform"/>
393           </ds:Transforms>
394           <ds:DigestMethod Algorithm=
395 "http://www.w3.org/2000/09/xmldsig#sha1"/>
396           <ds:DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</ds:DigestValue>
397         </ds:Reference>
398       </ds:SignedInfo>
399       <ds:SignatureValue>DeadBeef</ds:SignatureValue>
400     <ds:KeyInfo>
401       <wsse:SecurityTokenReference>
402         <wsse:Reference URI="#CertAssociatedWithSigningKey"/>
403       </wsse:SecurityTokenReference>
404     </ds:KeyInfo>
405   </ds:Signature>
406   </wsse:Security>
407 </S11:Header>
408 <S11:Body>
409   some items
410 </S11:Body>
411 </S11:Envelope>
412 --BoundaryStr
413 Content-Type: image/png
414 Content-ID: <bar>
415 Content-Transfer-Encoding: base64
416 the image
```

417 4.5 Encryption

418 A SwA attachment may be encrypted for confidentiality protection, protecting either the MIME part content
419 including selected MIME headers, or only the MIME part content.

420 This is done using XML Encryption to encrypt the attachment, placing the resulting cipher text in the

421 updated attachment body replacing the original content, and placing a new <xenc:EncryptedData>
422 element in the <wsse:Security> header. An <xenc:CipherReference> MUST link the
423 <xenc:EncryptedData> element with the cipher data.

424 The key used for encryption MAY be conveyed using an <xenc:EncryptedKey> element in the
425 <wsse:Security> header. In this case the <xenc:ReferenceList> element in the <xenc:EncryptedKey>
426 element MUST contain an <xenc:DataReference> with a URI attribute specifying the
427 <xenc:EncryptedData> element in the <wsse:Security> header corresponding to the attachment.

428 When the same <xenc:EncryptedKey> corresponds to multiple <xenc:EncryptedData> elements, the
429 <xenc:ReferenceList> in the <xenc:EncryptedKey> element SHOULD contain an <xenc:DataReference>
430 for each <xenc:EncryptedData> element, both for attachments and encrypted items in the primary SOAP
431 envelope. References should be ordered to correspond to ordering of the security header elements.

432 When an <xenc:EncryptedKey> element is not used when encrypting an attachment, then the
433 <xenc:EncryptedData> element MAY contain a <ds:KeyInfo> element to specify a key as outlined in the
434 SOAP Message Security standard. Different deployments may have different requirements on how keys
435 are referenced. When an <xenc:EncryptedKey> element is used the <xenc:EncryptedData> element
436 MUST NOT contain a <ds:KeyInfo> element.

437 When an attachment is encrypted, no <xenc:ReferenceList> element is placed as a direct child of the
438 <wsse:Security> header, since the <xenc:EncryptedData> element is present in the header, eliminating
439 the need for this reference. Although the SOAP Message Security standard recommends the use of
440 <xenc:ReferenceList>, this is only necessary when the <xenc:EncryptedData> element is not present in
441 the <wsse:Security> header. (As mentioned, when the key is conveyed in an <xenc:EncryptedKey>
442 element, then this element will have a ReferenceList Reference to the <xenc:EncryptedData> element).

443 **Note: The same CID is used to refer to the attachment before encryption and after.**
444 **This avoids the need to rewrite references to the attachment, avoiding issues**
445 **related to generating unique CIDs and relating to preserving the correspondence to**
446 **the original WSDL definition.**

447 **4.5.1 MIME Part CipherReference**

448 This profile requires that <xenc:EncryptedData> elements corresponding to encrypted SwA attachments
449 use a <xenc:CipherReference> to refer to the cipher text, to be conveyed in the attachment. Upon
450 encryption the MIME part attachment content is replaced with the encoded cipher text.

451 The <xenc:CipherReference> MUST have a <xenc:Transforms> child element. This element MUST have
452 a <ds:Transform> child having an Algorithm attribute with a URI value specifying the Content-Only MIME
453 Part Reference Transform. This transform explicitly indicates that when dereferencing the MIME part
454 reference that only the MIME part content is to be used as the cipher value.

455 The <xenc:CipherReference> MUST NOT contain a transform used for a transfer encoding purpose (e.g.
456 the base64 transform). Transfer encoding is left to the MIME layer, as noted in section 2.

457 **4.5.2 Encryption Processing Rules**

458 The order of the following steps is not normative, although the result should be the same as if this order
459 were followed.

460 1. When encrypting both attachments and primary SOAP envelope content using the same key, perform
461 the attachment processing first.

462 **Note: The SOAP Message Security standard states that elements should be**
463 **prepended to the security header. This processing rule supports putting the**
464 **<xenc:EncryptedData> element first in the header with <xenc:EncryptedKey> and**
465 **tokens following. Thus, a receiver should be able to process the**

466 **<xenc:EncryptedKey> before the <xenc:EncryptedData> element for the**
467 **attachment.**

- 468 2. Encrypt the attachment part using XML Encryption, according to the rules of XML Encryption. Encrypt
469 either the attachment including content and selected MIME headers or only the attachment content.
- 470 3. Set the <xenc:EncryptedData> Type attribute value to a URI that specifies adherence to his profile and
471 that specifies what was encrypted (MIME content or entire MIME part including headers). The following
472 URIs MUST be used for this purpose:

- 473 • Content Only: <http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-Only>.
- 474
- 475 • Content and headers: <http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Complete>
- 476

477 **Note: When this document is finalized these URLs will be updated, replacing XX**
478 **values and possibly making other changes. Note that these URLs should match the**
479 **related transforms apart from -Transform.**

- 480 4. Set the <xenc:EncryptedData> MimeType attribute to match the attachment MIME part Content-Type
481 header before encryption when Content-Only URI is specified for the Type attribute value. The
482 MimeType attribute value may also be set when the AttachmentComplete Type attribute value is
483 specified.
- 484 5. Optionally set the <xenc:EncryptedData> Encoding attribute to reflect the attachment content
485 encoding, as visible to the security layer at the time of encryption. This is advisory information to the
486 decryption security layer. It should be understood that this has no relation with the actual encoding that
487 could be performed independently by the MIME layer later for transfer purposes.
- 488 6. Set the <xenc:EncryptedData> <xenc:CipherReference> to the same reference URL for the
489 attachment that was used before encryption . This MUST be a CID scheme URL referring to the
490 attachment part Content-ID. Ensure this MIME header is in the part conveying the cipher data after
491 encryption.
- 492 7. Include the Content-Only MIME Part Reference Transform in the <xenc:CipherReference>
493 <xenc:Transforms> list.
- 494 8. Prepend the <xenc:EncryptedData> element to the <wsse:Security> SOAP header block. Do NOT add
495 a <xenc:ReferenceList> element to the SOAP header block (even though recommended by SOAP
496 Message Security).
- 497 9. Update the attachment MIME part, replacing the original content with the cipher text generated by the
498 XML Encryption step.
- 499 10. Update the attachment MIME part header MIME Content-Type and Content-Length appropriate to the
500 cipher data.

501 **4.5.3 Decryption Processing Rules**

502 The <xenc:CipherReference> URL MUST be a URL that refers to the MIME part containing the cipher
503 text, and must also correspond to the reference value of the original attachment that was encrypted. This
504 MUST be a CID scheme URL.

505 Decryption may be initiated upon locating the <xenc:EncryptedData> element in the <wsse:Security>
506 header.

507 The following decryption steps must be performed so that the result is as if they were performed in this
508 order:

- 509 1. Extract the cipher text from the attachment referenced by the <xenc:CipherReference> URL attribute.

510 The MIME Part CipherReference Transform defined in this profile indicates that the MIME part content
511 is extracted.

- 512 2. Decrypt the cipher text using the information present in the appropriate <xenc:EncryptedData> element
513 and possibly other out of band information, according to the XML Encryption Standard.
- 514 3. If the <xenc:EncryptedData>Type attribute indicates that selected MIME headers were encrypted, then
515 those MIME headers MUST be replaced by the result of decryption, as well as the MIME part content.
- 516 4. If the <xenc:EncryptedData>Type attribute indicates that only the content of the MIME part was
517 encrypted, then the cipher text content of the attachment part MUST be replaced by the result of
518 decryption. In this case the MIME part Content-Type header value MUST be replaced by the
519 <xenc:EncryptedData> MimeType attribute value.
- 520 5. If the <xenc:EncryptedData> Encoding attribute is present then the decryption security layer may pass
521 this advisory information to the application.

522 4.5.4 Example

523 This example shows encryption of the primary SOAP envelope body as well as an attachment using a
524 single symmetric key conveyed using an EncryptedKey element.

```
525 Content-Type: multipart/related; boundary="BoundaryStr" type="text/xml"
526 --BoundaryStr
527 Content-Type: text/xml

528 <S11:Envelope
529   xmlns:S11="http://schemas.xmlsoap.org/soap/envelope/"
530   xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
531 wsswssecurity-secext-1.0.xsd"
532   xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
533   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"

534   <S11:Header>
535     <wsse:Security>

536       <wsse:BinarySecurityToken wsu:Id="Acert"
537         EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
538 200401-wss-soap-message-security-1.0#Base64Binary"
539         ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
540 200401-wss-x509-token-profile-1.0#x509v3">
541         ...
542       </wsse:BinarySecurityToken>

543       <xenc:EncryptedKey Id='EK'>
544         <EncryptionMethod
545           Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
546         <ds:KeyInfo Id="keyinfo">
547           <wsse:SecurityTokenReference>
548             <ds:X509Data>
549               <ds:X509IssuerSerial>
550                 <ds:X509IssuerName>
551                   DC=ACMECorp, DC=com
552                 </ds:X509IssuerName>
553                 <ds:X509SerialNumber>12345678</X509SerialNumber>
554               </ds:X509IssuerSerial>
555             </ds:X509Data>
556           </wsse:SecurityTokenReference>
557         </ds:KeyInfo>
558         <CipherData><CipherValue>xyzabc</CipherValue></CipherData>
559         <ReferenceList>
```

```

560     <DataReference URI='#EA' />
561     <DataReference URI='#ED' />
562   </ReferenceList>
563 </EncryptedKey>

564     <xenc:EncryptedData
565       Id='EA'
566       Type="http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-
567 profile-1.0#Attachment-Content-Only"
568       MimeType="image/png">
569     <xenc:EncryptionMethod
570       Algorithm='http://www.w3.org/2001/04/xmlenc#aes128-cbc' />
571     <xenc:CipherData>
572       <xenc:CipherReference URI=cid:bar">
573         <xenc:Transforms>
574           <ds:Transform Algorithm="http://docs.oasis-
575 open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-
576 Only-Transform"/>
577         </xenc:Transforms>
578       </xenc:CipherReference>
579     </xenc:CipherData>
580   </xenc:EncryptedData>

581 </wsse:Security>
582 </S11:Header>
583 <S11:Body>
584   <xenc:EncryptedData Id='ED'
585     <xenc:EncryptionMethod
586       Algorithm='http://www.w3.org/2001/04/xmlenc#aes128-cbc' />
587     <xenc:CipherData>
588       <xenc:CipherValue>DEADBEEF</xenc:CipherValue>
589     </xenc:CipherData>
590   </xenc:EncryptedData>
591 </S11:Body>
592 </S11:Envelope>
593 --BoundaryStr
594 Content-Type: application/octet-stream
595 Content-ID: <bar>
596 Content-Transfer-Encoding: binary
597 BinaryCipherData

```

598 4.6 Signing and Encryption

599 When portions of content are both signed and encrypted, there is possible confusion as to whether
600 encrypted content need first be decrypted before signature verification. This confusion can occur when
601 the order of operations is not clear [[DecryptT](#)]. This problem may be avoided with SOAP Message Security
602 for SwA attachments when attachments and corresponding signatures and encryptions are targeted for a
603 single SOAP recipient (actor). The SOAP Message Security standard explicitly states that there may not
604 be two <wsse:Security> headers targeted at the same actor, nor may there be two headers without a
605 designated actor. In this case the SOAP Message Security and SwA profile processing rules may
606 eliminate ambiguity since each signing or encryption produces an element in the <wsse:Security> header,
607 and these elements are ordered. (Signing produces <ds:Signature> elements and encryption produces
608 <xenc:EncryptedData> elements).

609 If an application produces different <wsse:Security> headers targeted at different recipients, these are
610 processed independently by the recipients. Thus there is no need to correlate activities between distinct
611 headers – the order is inherent in the SOAP node model represented by the distinct actors.

5 References

612

- 613 **[CHARSETS]** Character sets assigned by IANA. See <ftp://ftp.isi.edu/in->
614 [notes/iana/assignments/character-sets](ftp://ftp.isi.edu/in-notes/iana/assignments/character-sets).
- 615 **[DecryptT]** M. Hughes et al, "Decryption Transform for XML Signature", W3C Recommendation 10
616 December 2002 <http://www.w3.org/TR/xmlenc-decrypt/>.
- 617 **[MTOM]** *Work in Progress – subject to change*. SOAP Message Transmission Optimization
618 Mechanism, W3C Candidate Recommendation 26 August 2004,
619 <http://www.w3.org/TR/soap12-mtom/>.
- 620 **[RFC2045]** Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message
621 Bodies, IETF RFC 2045, November 1996, <http://www.ietf.org/rfc/rfc2045.txt>.
- 622 **[RFC2046]** Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types, IETF RFC 2046,
623 November 1996, <http://www.ietf.org/rfc/rfc2046.txt>.
- 624 **[RFC2047]** Multipurpose Internet Mail Extensions (MIME) Part Three: Message Header Extensions
625 for Non-ASCII Text, IETF RFC 2047, November 1996, <http://www.ietf.org/rfc/rfc2047.txt>.
- 626 **[RFC2048]** Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures,
627 <http://www.ietf.org/rfc/rfc2048.txt>.
- 628 **[RFC2049]** Multipurpose Internet Mail Extensions(MIME) Part Five: Conformance Criteria and
629 Examples, <http://www.ietf.org/rfc/rfc2049.txt>.
- 630 **RFC2119]** S. Bradner, Key words for use in RFCs to Indicate Requirement Levels, IETF RFC 2119,
631 March 1997, <http://www.ietf.org/rfc/rfc2119.txt>.
- 632 **[RFC2392]** E. Levinson, *Content-ID and Message-ID Uniform Resource Locators*, IETF RFC 2392,
633 <http://www.ietf.org/rfc/rfc2392.txt>
- 634 **[RFC2557]** MIME Encapsulation of Aggregate Documents, such as HTML (MHTML), IETF RFC
635 2557, March 1999, <http://www.ietf.org/rfc/rfc2557.txt>.
- 636 **[RFC2633]** Ramsdell B., "S/MIME Version 3 Message Specification", Standards Track RFC 2633,
637 June 1999. <http://www.ietf.org/rfc/rfc2633.txt>
- 638 **[RFC2822]** Internet Message Format, IETF RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>.
- 639 **[SECGLO]** Informational RFC 2828, "Internet Security Glossary," May 2000.
- 640 **[SOAP11]** W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May 2000.
- 641 **[SWA]** W3C Note, "SOAP Messages with Attachments", 11 December 2000,
642 <http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211> .
- 643 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic
644 Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998,
645 <http://www.ietf.org/rfc/rfc2396.txt>.
- 646 **[WS-I-AP]** *Final Material* Attachments Profile Version 1.0, 2004-08-24, [http://www.ws-](http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html)
647 [i.org/Profiles/AttachmentsProfile-1.0.html](http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html)
- 648 **[WSS-Sec]** A. Nadalin et al., Web Services Security: SOAP Message Security 1.0 (WS-Security
649 2004), OASIS Standard 200401, March 2004, [http://docs.oasis-](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)
650 [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)
- 651 **[XML-Schema]** W3C Recommendation, "XML Schema Part 1: Structures," 2 May 2001,
652 <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>.
- 653 W3C Recommendation, "XML Schema Part 2: Datatypes," 2 May 2001,
654 <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>.
- 655 **[XPath]** W3C Recommendation, "XML Path Language", 16 November 1999,
656 <http://www.w3.org/TR/xpath>.

657 A. Acknowledgments

658 The editors would like to acknowledge the contributions of the OASIS WSS Technical Committee, whose
659 voting members at the time of publication were:

- 660 • Gene Thurston, AmberPoint
- 661 • Frank Siebenlist, Argonne National Laboratory
- 662 • Hal Lockhart, BEA Systems, Inc.
- 663 • Corinna Witt, BEA Systems, Inc.
- 664 • Thomas DeMartini, ContentGuard
- 665 • Guillermo Lao, ContentGuard
- 666 • Merlin Hughes, Cybertrust
- 667 • Sam Wei, Documentum
- 668 • Tim Moses, Entrust
- 669 • Carolina Canales-Valenzuela, Ericsson
- 670 • Dana Kaufman, Forum Systems, Inc.
- 671 • Toshihiro Nishimura, Fujitsu
- 672 • Kefeng Chen, GeoTrust
- 673 • Irving Reid, Hewlett-Packard
- 674 • Kojiro Nakayama, Hitachi
- 675 • Paula Austel, IBM
- 676 • Derek Fu, IBM
- 677 • Maryann Hondo, IBM
- 678 • Kelvin Lawrence, IBM
- 679 • Michael McIntosh, IBM
- 680 • Anthony Nadalin, IBM
- 681 • Nataraj Nagaratnam, IBM
- 682 • Ron Williams, IBM
- 683 • Ramanathan Krishnamurthy, IONA
- 684 • Kate Cherry, Lockheed Martin
- 685 • Paul Cotton, Microsoft Corporation
- 686 • Vijay Gajjala, Microsoft Corporation
- 687 • Martin Gudgin, Microsoft Corporation
- 688 • Chris Kaler, Microsoft Corporation
- 689 • Rich Levinson, Netegrity, Inc.
- 690 • Jeff Hodges, Neustar, Inc.

- 691 • Frederick Hirsch, Nokia
- 692 • Abbie Barbir, Nortel Networks
- 693 • Lloyd Burch, Novell
- 694 • Steve Anderson, OpenNetwork
- 695 • Vamsi Motukuru, Oracle
- 696 • Ramana Turlapati, Oracle
- 697 • Chong-Jen Hsu, PeopleSoft
- 698 • Prateek Mishra, Principal Identity
- 699 • Ben Hammond, RSA Security
- 700 • Rob Philpott, RSA Security
- 701 • Martijn de Boer, SAP
- 702 • Blake Dournaee, Sarvega
- 703 • Coumara Radja, Sarvega
- 704 • Pete Wenzel, SeeBeyond Technology Corporation
- 705 • Ronald Monzillo, Sun Microsystems
- 706 • Jan Alexander, Systinet
- 707 • Symon Chang, Tibco
- 708 • John Weiland, US Dept of the Navy
- 709 • Phillip Hallam-Baker, Verisign
- 710 • Maneesh Sahu, Westbridge Technology

B. Revision History

Rev	Date	By Whom	What
1	05/25/04	Frederick Hirsch	Initial version, put draft proposal into profile format.
2	05/26/04	Frederick Hirsch	Editorial and namespace suggestions from Michael McIntosh. Added rationale for SwA support to introduction. Completely rewrote processing rules for encryption and decryption.
3	05/28/04	Frederick Hirsch	Rewrote signature section, fixed cid references and Content-IDs, added examples.
4	06/12/04	Frederick Hirsch	Added Decrypt Transform section, added All-Attachments-Complete transform, changed MIME reference to v3, minor editorial changes.
5	07/07/04	Frederick Hirsch	Removed Decrypt transform material, since it is generally not needed and the approach had issues. Reorganized signatures section. Eliminated incorrect All-Attachments-Complete transform and replaced with discussion of attachment insertion threat. Clarified that only one wsse:Security header per actor/role minimizes signing, encryption confusion possibility. Added section for MIME Part CipherReference Transform. Editorial fixes.
6	07/14/04	Frederick Hirsch	<p>** Allow use of Content-Location, consistent with SwA.</p> <p>** Proposed update to signature Content-Transfer-Encoding processing rules. Needs review.</p> <p>Revised section on MIME canonicalization, added section on XML attachments. Only support SOAP 1.1. Clarified introduction. Added MTOM and additional MIME references. (Issue 297 should be closed – removed section on decryption transform and updated section on signing and encryption in version 5) Issue 303 – fixed, (see 3.2.4 example), Issue 306 – revised section on MIME canonicalization to close this issue. Issue 307 – revised to refer to SOAP 1.1 only, added section on XML attachments, defined MTOM and added reference. Editorial fixes.</p>

Rev	Date	By Whom	What
7	07/30/04	Frederick Hirsch	Incorporate feedback from WS-I BSP. Limit MIME headers included in signature or encryption to those listed in profile. Clarify MIME layering approach. Remove processing rules associated with Content-Transfer-Encoding. Editorial correction throughout document to allow both CID and Content-Location references to attachments. Editorial revision to pull attachment referencing and reference transforms into section applicable to both signatures and encryption. Incorporated feedback from Pete Wenzel and Toshihiro Nishimura – separate URL for transform and encryption type, used Content-Only reference transform for Cipherdata as well.
8	08/23/04	Frederick Hirsch	Address issue 312 by clarifying use of Reference within EncryptedData element to EncryptedData for attachment when EncryptedKey is used. Processing rule related to encryption of both attachment and primary SOAP envelope items. (http://www.oasis-open.org/archives/wss/200408/msg00046.html) Changed encryption example to show encryption of both primary SOAP envelop body and attachment. Include EncryptionMethod, addressing issue 309. Fix Transforms namespace to be xenc for within xenc:CipherReference (http://www.oasis-open.org/archives/wss/200408/msg00048.html)
9	09/02/04	Frederick Hirsch	Clarify that XML attachments are opaque and remove text about XML canonicalization of attachment content. Fix typo at line 356, should state that no KeyInfo should be in EncryptedData element when EncryptedKey is used. Clarify that cipher data is base64 encoded octet stream and require CipherReference base64 transform. Revise MIME headers to be included in Attachment-Complete Reference, for signature protection. Allow continuations for these MIME headers.

Rev	Date	By Whom	What
10	10/02/04	Frederick Hirsch	<p>Proposed resolutions for WSS issue-list items:</p> <p>Issue 326 part 1 – corrected case of Content-ID throughout document.</p> <p>Issue 326 part 2 - : Clarify MIME header name case, Resolution to use case per MIME specifications. See 4.3.1 item 4.</p> <p>Issue 326 part 3- Clarify transform handling of MIME parameter quoting. Retain quoting, if any, as is. Resolution in 4.3.1 item 7.</p> <p>Issue 326 part 4 - Address RFC 2047 encoding. Require transform to perform RFC2047 decoding as needed. Resolution in 4.3.1, items 4-7.</p> <p>Issue 329 part 1 – Strip or compress white space. No change made apart apart from preserve all whitespace in quoted strings, 4.3.1. item 10.</p> <p>Issue 329 part 2 – Order header processing alphabetically. Resolution in 4.3.1 item 3 and 4.2.2.</p> <p>Issue 329 part 3 – Show all ds:Signature elements in example in 4.3.6.</p>
11	10/02/04	Frederick Hirsch	Issue 326, 329 – revision of section 4.3.1 based on feedback from Dana Kaufman and Forum Systems.
12	10/21/04	Frederick Hirsch	<p>Allow cipher data to be binary data, and not use base64 transform in this case. Clarify that for base64 encoded cipher data transform or other means should be used to convey this information. Updated 4.4.1 through 4.4.4.</p> <p>Quoted “text/xml” in examples in 4.3.6, 4.4.4 to resolve issue 325.</p>
13	10/29/04	Frederick Hirsch	<p>Replace “7-bit” with “binary” in example 4.4.4</p> <p>Add clarification to sections 4.2.1 and 4.2.2 that MIME canonicalization is to be associated with the transforms, as defined in 4.3.1 and 4.3.2.</p>
14	11/15/04	Frederick Hirsch	<ol style="list-style-type: none"> 1. Only allow CID references for WS-Security references, for simplicity and interoperability. 2. Constrain statement on RFC2047 encoding in section 4.4.1, #6. 3. Clarify use of <xenc:EncryptedData> MimeType attribute in 4.5.2, #4. (Issue 345, #1) 4. Add statement from interop document regarding MIME boundary for primary SOAP envelope 5. Editorial changes to make MAY/MUST/SHOULDs capitalized where possible, other editorial fixes.

Rev	Date	By Whom	What
15	12/06/04	Frederick Hirsch	<p>Explicitly allow optional use of Encryption Encoding attribute. (Section 4.5.2 #5; Issue 341)</p> <p>Remove base64 transform material, clarify relationship to MIME layer transform encoding. (Sections 4.4.4, 4.4.5, 4.5.1, 4.5.2, 4.5.3; Issue 344)</p> <p>Add clarification that Content-ID header value <> included in Attachment-Complete transform for signing. (Section 4.4.1, #7)</p> <p>Editorial cleanup. Add KeyInfo to example 4.4.6.</p>
cd-01	01/07/05	Frederick Hirsch	Change to Committee Draft - 01

712 **C. Notices**

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

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

724 **Copyright © OASIS Open 2004. All Rights Reserved.**

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

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

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

739 JavaScript is a trademark or registered trademark of Sun Microsystems, Inc. in the United States and
740 other countries.