



# Web Services Security SOAP Messages with Attachments (SwA) Profile 1.0

OASIS Draft 19, 16 May 2005

## Document identifier:

wss-swa-profile-1.0-draft-19

## Location:

[http://www.oasis-open.org/committees/documents.php?wg\\_abbrev=wss](http://www.oasis-open.org/committees/documents.php?wg_abbrev=wss)

## Editors:

Frederick Hirsch, Nokia

## Contributors:

Michael Hu, Actional  
Maneesh Sahu, Actional  
Thomas DeMartini, ContentGuard  
Dale Moberg, Cyclone Commerce  
Dana S. Kaufman, Forum Systems, Inc  
Dan Smiley, Forum Systems, Inc  
Michael McIntosh, IBM  
Frederick Hirsch, Nokia  
Jerry Schwarz, Oracle  
Blake Dournaee, Sarvega, Inc.  
Pete Wenzel, SeeBeyond

## Abstract:

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

## Status:

This is a Draft and has no standing.

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

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 web page for the WSS TC (<http://www.oasis-open.org/committees/wss/ipr.php>).

## 37 **Table of Contents**

38	1 Introduction.....	3
39	1.1 Notations and Terminology.....	4
40	1.1.1 Notational Conventions.....	4
41	1.1.2 Namespaces.....	4
42	1.1.3 Acronyms and Abbreviations.....	5
43	2 MIME Processing.....	6
44	3 XML Attachments.....	7
45	4 Securing SOAP With Attachments.....	8
46	4.1 Primary SOAP Envelope.....	8
47	4.2 Referencing Attachments.....	8
48	4.3 MIME Part Reference Transforms.....	8
49	4.3.1 Attachment-Content-Only Reference Transform.....	9
50	4.3.2 Attachment-Complete Reference Transform.....	9
51	4.4 Integrity and Data Origin Authentication .....	10
52	4.4.1 MIME header canonicalization.....	10
53	4.4.2 MIME Content Canonicalization.....	11
54	4.4.3 Protecting against attachment insertion threat.....	12
55	4.4.4 Processing Rules for Attachment Signing.....	12
56	4.4.5 Processing Rules for Attachment Signature Verification.....	12
57	4.4.6 Example Signed Message.....	14
58	4.5 Encryption.....	14
59	4.5.1 MIME Part CipherReference.....	15
60	4.5.2 Encryption Processing Rules.....	15
61	4.5.3 Decryption Processing Rules.....	16
62	4.5.4 Example.....	17
63	4.6 Signing and Encryption.....	18
64	5 References.....	19

---

# 1 Introduction

65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107

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.

Goals of this profile include the following:

- Enable those who choose to use SwA to secure these messages, including chosen attachments, using SOAP Message Security
- Allow the choice of securing MIME header information exposed to the SOAP layer, if desired.
- Do not interfere with MIME transfer mechanisms, in particular, allow MIME transfer encodings to change to support MIME transfer, despite support for integrity protection.
- Do not interfere with the SOAP processing model – in particular allow SwA messages to transit SOAP intermediaries.

Non-goals include:

- Provide guidance on which of a variety of security mechanisms are appropriate to a given application. The choice of transport layer security (e.g. SSL/TLS), S/MIME, application use of XML Signature and XML Encryption, and other SOAP attachment mechanisms (MTOM) is explicitly out of scope. This profile assumes a need and desire to secure SwA using SOAP Message security.
- Outline how different security mechanisms may be used in combination.
- Enable persisting signatures. It may be possible depending on the situation and measures taken, but is not discussed in this profile.
- Support signing and/or encryption of portions of attachments. This is not supported by this profile, but is not necessarily precluded. Application use of XML Signature and XML Encryption may be used to accomplish this. SOAP Message security may also support this in some circumstances, but this profile does not address or define such usage.

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.

108 Use of this profile is intended to be independent of S/MIME when S/MIME technology is used to sign  
109 and/or encrypt attachment content (MIME parts that do not contain the primary SOAP envelope). When  
110 enveloped S/MIME is used, a new attachment is created that can be treated as opaque by this profile,  
111 e.g. An application/pkcs7-mime type attachment. When the signed-only multi-part/signed S/MIME format  
112 is used, then there will be two attachment parts, the part that has been signed and the signature. In this  
113 case it is necessary for any WS-Security encryption to be removed before S/MIME validation occurs. In  
114 addition, any canonicalization must be consistent with S/MIME canonicalization. This is the case if the part  
115 is not of an XML type, since this profile requires MIME canonicalization. If it is an xml type then it is  
116 unlikely to be true, since this profile also requires XML canonicalization. For this reason, in this case an  
117 application/pkcs7-mime format signature is recommended. Regardless, there is no explicit interaction  
118 between the security mechanisms outlined in this specification and any S/MIME used with associated  
119 attachments.

## 120 1.1 Notations and Terminology

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

### 122 1.1.1 Notational Conventions

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

126 `Listings of productions or other normative code appear like this.`

127 `Example code listings appear like this.`

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

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

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

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

### 139 1.1.2 Namespaces

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

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

Prefix	Namespace
S11	<a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a>

wsse	<a href="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd</a>
wsu	<a href="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd</a>
wsswa	<a href="http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0.xsd">http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0.xsd</a>

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

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

## 150 1.1.3 Acronyms and Abbreviations

151 The following (non-normative) table defines acronyms and abbreviations for this document, beyond those  
152 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

154 This profile is concerned with the securing of SOAP messages with attachments, attachments that are  
155 conveyed as MIME parts in a multi-part MIME message as outlined in SOAP Messages with Attachments  
156 [SwA]. This involves two processing layers, SOAP messaging and MIME transfer. This specification  
157 defines processing of a merged SOAP and MIME layer, in order to meet SwA security requirements. It  
158 relies on an underlying MIME transfer layer that allows changes to MIME transfer encoding as a message  
159 transits MIME nodes. This profile does not impose restrictions on that MIME transfer layer apart from  
160 aspects that are exposed to the SOAP processing layer. Likewise, this profile does not restrict the SOAP  
161 processing model, including use of SOAP intermediaries, allowing SOAP Messages with Attachments to  
162 transit SOAP nodes.

163 To accommodate the ability to secure attachment headers that are exposed to the SOAP message layer  
164 and application, this profile does not assume a strict protocol layering of MIME, SOAP and application.  
165 Rather, this profile allows a SOAP sender to create a primary SOAP envelope as well as attachments to  
166 be sent with the message. It is up to the application which, if any, of the attachments are referenced from  
167 SOAP header and/or body blocks. The application may be aware of, and concerned with, certain aspects  
168 of the attachment MIME representation, including Content-Type and Content-Length headers, to give two  
169 examples. Due to this concern, the application may choose to secure these exposed headers. This does  
170 not mean, however, that the application and SOAP layer are aware or concerned with all MIME headers  
171 used for MIME transit, in particular issues related to transfer encoding. The expectation is that the MIME  
172 processing layer of the sender and receiver will handle transfer encoding issues, hiding this detail from the  
173 processing layer associated with this profile. As a result, this specification focuses on those aspects of  
174 MIME processing that are exposed and of concern to higher protocol layers, while ignoring MIME transit  
175 specific details.

176 This model has two implications. First, it means that certain aspects of MIME processing, such as transfer  
177 encoding processing, are out of scope of the profile and do not need to be addressed. Secondly, it means  
178 that many of the MIME headers are also out of scope of the profile and the profile does not support  
179 integrity protection of these headers, since they are expected to change. If more security protection is  
180 required then it must occur by other means, such as with a protocol layer below the MIME layer, for  
181 example transport security (with the understanding that such security may not always apply end-end).

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

187

---

## 3 XML Attachments

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

191 Some attachments associated with the SOAP body may be targeted at the SOAP Ultimate Receiver along  
192 with the SOAP body and may be processed at the application layer along with the body. Others may be  
193 targeted at intermediaries. How attachments are to be processed and how these attachments are  
194 referenced from SOAP header and body blocks, if at all, is dependent on the application. In many cases  
195 the attachment content may not need to be processed as XML as the message traverses intermediaries.

196 Requiring canonicalization of XML attachments is undesirable, both due to the potential ambiguities  
197 related to the canonicalization context of the attachment (e.g. Is it an independent XML document, a  
198 portion of the primary SOAP envelope, etc) as well as the universal performance impact of such a  
199 canonicalization requirement.

200 XML processing may not be required for XML media type MIME attachments until application layer  
201 processing is performed. For this reason the SOAP message layer does not need to perform XML  
202 canonicalization or parsing for such attachments and SOAP Message layer security may treat these  
203 attachments as text. In those cases where XML processing of an attachment at an intermediary cannot be  
204 avoided, due to the nature of the implementation, a sender may choose to encode the attachment so that  
205 it is not recognized as XML.

206 This profile assumes that SOAP attachments (not including the root part containing the primary SOAP  
207 envelope) need not be processed as XML at the SOAP messaging layer, so do not require SOAP  
208 canonicalization or XML parsing and may be treated as opaque data by the SOAP Message Security layer  
209 security processing. In those cases where there is a concern of unintended XML processing of XML  
210 attachments by intermediaries, the sender SHOULD encode the attachment so that it is not processed as  
211 XML.

212 MIME part canonicalization (as described below) is required or XML attachments to enable SOAP  
213 Message Security signatures that are stable despite MIME transfer processing.

---

## 214 4 Securing SOAP With Attachments

215 Attachments may be associated with SOAP messages, as outlined in SOAP Messages with Attachments  
216 [SwA]. This profile defines how such attachments may be secured for integrity and confidentiality using the  
217 OASIS Web Services Security: SOAP Message Security standard. This does not preclude using other  
218 techniques. The requirements in this profile only apply when securing SwA attachments explicitly  
219 according to this profile.

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

### 223 4.1 Primary SOAP Envelope

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

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

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

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

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

```
235 --xyl  
236 Content-Type: text/xml; charset=utf-8  
237 Content-ID: <foo>  
  
238 <?xml version='1.0' ?>  
239 <s11:Envelope xmlns:s11="http://schemas.xmlsoap.org/soap/envelope/" />
```

### 240 4.2 Referencing Attachments

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

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

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

### 251 4.3 MIME Part Reference Transforms

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

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

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

259 The input of each of these transforms is an octet stream, as defined in XML Security [XML-Sig].

### 260 4.3.1 Attachment-Content-Only Reference Transform

261 The Attachment-Content-Only transform indicates that only the content of a MIME part is referenced. This  
262 transform MUST be identified using the URI value: [http://docs.oasis-open.org/wss/2004/XX/oasis-  
263 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).

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

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

268 The octet stream input to this transform is the entire content of the MIME attachment associated with the  
269 CID, including all the MIME headers and attachment content, as represented in the MIME part containing  
270 the attachment.

271 The output of the transform is an octet stream consisting of the MIME canonicalization of the attachment  
272 content. All of the MIME headers associated with the MIME part are ignored and not included in the output  
273 octet stream. The MIME canonicalization of content is described elsewhere in this specification.

### 274 4.3.2 Attachment-Complete Reference Transform

275 The Attachment-Complete transform indicates that both the content and selected headers of the MIME  
276 part are referenced. This transform MUST be identified using the URI value: [http://docs.oasis-  
277 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).

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

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

- 282 • Content-Description
- 283 • Content-Disposition
- 284 • Content-ID
- 285 • Content-Location
- 286 • Content-Type

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

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

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

294 The octet stream input to this transform is the entire content of the MIME attachment associated with the  
295 CID, including all the MIME headers and attachment content, as represented in the MIME part containing  
296 the attachment.

297 The output of the transform is an octet stream consisting of concatenation of the MIME canonicalized  
298 MIME headers selected by the transform followed by the concatenation of the MIME canonicalization of  
299 the attachment content. The MIME canonicalization of headers and content are described elsewhere in  
300 this specification.

## 301 **4.4 Integrity and Data Origin Authentication**

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

### 309 **4.4.1 MIME header canonicalization**

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

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

- 315 1. The transform **MUST** process MIME headers before the MIME content.
- 316 2. The transform **MUST** only process MIME headers that are explicitly present in the attachment part and  
317 are listed in the Attachment-Complete transform section of this specification, except that a MIME part  
318 without a Content-Type header **MUST** be treated as having a Content-Type header with the value  
319 "Content-Type: text/plain; charset=us-ascii". MIME headers not listed in the Attachment-Complete  
320 transform section of this specification are to be ignored by the transform.
- 321 3. The MIME headers **MUST** be processed by the Attachment-Complete transform in lexicographic order  
322 (ascending).
- 323 4. The MIME header names **MUST** be processed by the transform as having the case according to the  
324 MIME specifications (as shown in the Attachment-Complete section).
- 325 5. The MIME header values **MUST** be unfolded [RFC2822].
- 326 6. Any Content-Description MIME header containing RFC2047 encoding **MUST** be decoded [RFC2047].
- 327 7. When a Content-ID header is processed, the "<>" characters associated with the msg-id **MUST** be  
328 included in the transform input. The reason is that although semantically these angle bracket  
329 characters are not part of the msg-id (RFC 2822) they are a standard part of the header lexicographic  
330 representation. If these characters are not integrity protected then an attacker could remove them  
331 causing the CID transformation specified in RFC2392 to fail.
- 332 8. Folding whitespace in structured MIME headers (e.g. Content-Disposition, Content-ID, Content-  
333 Location, Content-Type) that is not within quotes **MUST** be removed. Folding whitespace in structured  
334 MIME headers that is within quotes **MUST** be preserved. Folding whitespace in unstructured MIME  
335 headers (e.g. Content-Description) **MUST** be preserved [RFC2822]. For example, whitespace  
336 immediately following the colon delimiter in the structured Content-Type header **MUST** be removed,  
337 but whitespace immediately following the colon delimiter in the unstructured Content-Description  
338 header **MUST** be preserved.

- 339 9. Comments in MIME header values MUST be removed [RFC2822].
- 340 10. Case-insensitive MIME header values (e.g. media type/subtype values and disposition-type values)  
341 MUST be converted to lowercase. Case-sensitive MIME header values MUST be left as is with  
342 respect to case [RFC2045].
- 343 11. Quoted characters other than double-quote and backslash ("\") in quoted strings in structured MIME  
344 headers (e.g. Content-ID) MUST be unquoted. Double-quote and backslash ("\") characters in quoted  
345 strings in structured MIME headers MUST be character encoded [RFC2822].
- 346 12. Canonicalization of a MIME header MUST generate a UTF-8 encoded octet stream containing the  
347 following: the MIME header name, a colon (":"), the MIME header value, and the result of  
348 canonicalizing the MIME header parameters in lexicographic order (ascending) as described below.
- 349 13. MIME header parameter names MUST be converted to lowercase [RFC2045].
- 350 14. MIME parameter values containing RFC2184 character set, language, and continuations MUST be  
351 decoded. The resulting canonical output MUST not contain the RFC2184 encoding [RFC2184].
- 352 15. Case-insensitive MIME header parameter values MUST be converted to lowercase. Case-sensitive  
353 MIME header parameter values MUST be left as is with respect to case [RFC2045].
- 354 16. Enclosing double-quotes MUST be added to MIME header parameter values that do not already  
355 contain enclosing quotes. Quoted characters other than double-quote and backslash ("\") in MIME  
356 header parameter values MUST be unquoted. Double-quote and backslash characters in MIME  
357 parameter values MUST be character encoded.
- 358 17. Canonicalization of a MIME header parameter MUST generate a UTF-8 encoded octet stream  
359 containing the following: a semi-colon (";"), the parameter name (lowercase), an equals sign ("="), and  
360 the double-quoted parameter value.
- 361 18. Each header MUST be terminated by a single CRLF pair, without any trailing whitespace.
- 362 19. The last header MUST be followed by a single CRLF and then the MIME content.

#### 363 4.4.2 MIME Content Canonicalization

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

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

374 MIME types are registered. This registration includes a section on "Canonicalization and Format  
375 Requirements" [RFC2048] and requires each MIME type to have a canonical representation.

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

380 MIME attachment parts (other than the part containing the primary SOAP envelope) that contain XML do  
381 NOT require XML Canonicalization according to this profile, given the rationale in the previous section on  
382 XML attachments. These parts MUST be MIME canonicalized according the MIME "text" part  
383 requirements. MIME part canonicalization must be performed before signature hash generation or

384 verification is performed. Signature validation requires an identical hash of content requiring consistent  
385 MIME part content.

### 386 **4.4.3 Protecting against attachment insertion threat**

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

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

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

### 396 **4.4.4 Processing Rules for Attachment Signing**

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

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

402 For each attachment Reference, perform the following steps:

- 403 1. MIME Part Canonicalize the content of the attachment, as appropriate to the MIME type of the part.
- 404 2. If MIME headers are to be included in the signature, MIME part canonicalize the headers listed in this  
405 profile as outlined above.
- 406 3. Determine the CID scheme URL to be used to reference the part and set the <ds:Reference> URL  
407 attribute value to this URL.
- 408 4. Include a <ds:Transforms> element in the <ds:Reference>. This <ds:Transforms> element MUST  
409 include a <ds:Transform> element with the Algorithm attribute having the URL value specified in this  
410 profile - either Attachment-Complete or Attachment-Content-Only, depending on what is to be  
411 included in the hash calculation. This MUST be the first transform listed. The <ds:Transform> element  
412 MUST NOT contain any transform for a MIME transfer encoding purpose (e.g. base64 encoding)  
413 since transfer encoding is left to the MIME layer as noted in section 2. This does not preclude the use  
414 of XML Transforms, including a base64 transform, for other purposes.
- 415 5. Extract the appropriate portion of the MIME part consistent with the selected transform.
- 416 6. Create the <ds:Reference> hash value as outlined in the W3C XML Digital Signature  
417 Recommendation.

### 418 **4.4.5 Processing Rules for Attachment Signature Verification**

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

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

- 423 1. Find the attachment corresponding to the <ds:Reference> URL attribute value. This value MUST  
424 correspond to the Content-ID for the attachment[SwA].

- 425 2. MIME Part Canonicalize the content of the attachment, as appropriate to the MIME type of the part.  
426 The MIME content to be MIME canonicalized MUST have had any transfer-encoding processed at the  
427 MIME layer before this step is performed.
- 428 3. If MIME headers were included in the signature, canonicalize the headers listed in this profile as  
429 outlined above.
- 430 4. Extract the appropriate portion of the MIME part according to the MIME Part Signature Transform  
431 value.
- 432 5. Calculate the reference hash and verify the reference.

## 4.4.6 Example Signed Message

```
434 Content-Type: multipart/related; boundary="BoundaryStr" type="text/xml"
435 --BoundaryStr
436 Content-Type: text/xml
437 <S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..."
438 xmlns:ds="..." xmlns:xenc="...">
439   <S11:Header>
440     <wsse:Security>
441       <wsse:BinarySecurityToken wsu:Id="CertAssociatedWithSigningKey"
442         EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
443 200401-wss-soap-message-security-1.0#Base64Binary"
444         ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
445 200401-wss-x509-token-profile-1.0#x509v3">
446         ...
447       </wsse:BinarySecurityToken>
448     <ds:Signature>
449       <ds:SignedInfo>
450         <ds:CanonicalizationMethod Algorithm=
451 'http://www.w3.org/2001/10/xml-exc-c14n#' />
452         <ds:SignatureMethod Algorithm=
453 'http://www.w3.org/2000/09/xmldsig#rsa-sha1' />
454         <ds:Reference URI="cid:bar">
455           <ds:Transforms>
456             <ds:Transform Algorithm="http://docs.oasis-
457 open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-
458 Only-Transform"/>
459           </ds:Transforms>
460           <ds:DigestMethod Algorithm=
461 "http://www.w3.org/2000/09/xmldsig#sha1"/>
462           <ds:DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk=</ds:DigestValue>
463         </ds:Reference>
464       </ds:SignedInfo>
465       <ds:SignatureValue>DeadBeef</ds:SignatureValue>
466     <ds:KeyInfo>
467       <wsse:SecurityTokenReference>
468         <wsse:Reference URI="#CertAssociatedWithSigningKey"/>
469       </wsse:SecurityTokenReference>
470     </ds:KeyInfo>
471   </ds:Signature>
472   </wsse:Security>
473 </S11:Header>
474 <S11:Body>
475   some items
476 </S11:Body>
477 </S11:Envelope>
478 --BoundaryStr
479 Content-Type: image/png
480 Content-ID: <bar>
481 Content-Transfer-Encoding: base64
482 the image
```

## 4.5 Encryption

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

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

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

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

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

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

503 When an attachment is encrypted, an <xenc:EncryptedData> element will be placed in the  
504 <wsse:Security> header. An <xenc:ReferenceList> element associated with this  
505 <xenc:EncryptedData> element may also be added, as recommended by WSS: SOAP Message Security.

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

#### 510 **4.5.1 MIME Part CipherReference**

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

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

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

#### 520 **4.5.2 Encryption Processing Rules**

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

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

525 **Note: The SOAP Message Security standard states that elements should be**  
526 **prepended to the security header. This processing rule supports putting the**  
527 **<xenc:EncryptedData> element first in the header with <xenc:EncryptedKey> and**  
528 **tokens following. Thus, a receiver should be able to process the**  
529 **<xenc:EncryptedKey> before the <xenc:EncryptedData> element for the**  
530 **attachment.**

531 2. Encrypt the attachment part using XML Encryption, according to the rules of XML Encryption. Encrypt  
532 either the attachment including content and selected MIME headers or only the attachment content.

533 When encryption includes MIME headers, only the headers listed in this specification for the Attachment-  
534 Complete Reference Transform (Section 4.3.2) are to be included in the encryption. If a header listed in  
535 the profile is present it MUST be included in the encryption. If a header is not listed in this profile, then it  
536 MUST NOT be included in the encryption.

537 3. Set the <xenc:EncryptedData> Type attribute value to a URI that specifies adherence to this profile and  
538 that specifies what was encrypted (MIME content or entire MIME part including headers). The following  
539 URIs MUST be used for this purpose:

- 540 • Content Only: [http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-  
541 1.0#Attachment-Content-Only](http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-Only).
- 542 • Content and headers: [http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-  
543 1.0#Attachment-Complete](http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Complete)

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

547 4. Set the <xenc:EncryptedData> MimeType attribute to match the attachment MIME part Content-Type  
548 header before encryption when Content-Only URI is specified for the Type attribute value. The  
549 MimeType attribute value may also be set when the AttachmentComplete Type attribute value is  
550 specified.

551 5. Optionally set the <xenc:EncryptedData> Encoding attribute to reflect the attachment content  
552 encoding, as visible to the security layer at the time of encryption. This is advisory information to the  
553 decryption security layer. It should be understood that this has no relation with the actual encoding that  
554 could be performed independently by the MIME layer later for transfer purposes.

555 6. Set the <xenc:EncryptedData> <xenc:CipherReference> to the same reference URL for the  
556 attachment that was used before encryption . This MUST be a CID scheme URL referring to the  
557 attachment part Content-ID. Ensure this MIME header is in the part conveying the cipher data after  
558 encryption.

559 7. Include the Content-Only MIME Part Reference Transform in the <xenc:CipherReference>  
560 <xenc:Transforms> list.

561 8. Prepend the <xenc:EncryptedData> element to the <wsse:Security> SOAP header block and then  
562 prepend the associated optional <xenc:ReferenceList> element.

563 9. Update the attachment MIME part, replacing the original content with the cipher text generated by the  
564 XML Encryption step.

565 10. Update the attachment MIME part header MIME Content-Type and Content-Length appropriate to the  
566 cipher data.

### 567 **4.5.3 Decryption Processing Rules**

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

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

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

- 575 1. Extract the cipher text from the attachment referenced by the <xenc:CipherReference> URL attribute.  
576 The MIME Part CipherReference Transform defined in this profile indicates that the MIME part content  
577 is extracted.
- 578 2. Decrypt the cipher text using the information present in the appropriate <xenc:EncryptedData> element  
579 and possibly other out of band information, according to the XML Encryption Standard.
- 580 3. If the <xenc:EncryptedData>Type attribute indicates that selected MIME headers were encrypted, then  
581 those MIME headers MUST be replaced by the result of decryption, as well as the MIME part content.
- 582 4. If the <xenc:EncryptedData>Type attribute indicates that only the content of the MIME part was  
583 encrypted, then the cipher text content of the attachment part MUST be replaced by the result of  
584 decryption. In this case the MIME part Content-Type header value MUST be replaced by the  
585 <xenc:EncryptedData> MimeType attribute value.
- 586 5. If the <xenc:EncryptedData> Encoding attribute is present then the decryption security layer may pass  
587 this advisory information to the application.

#### 588 4.5.4 Example

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

```

591 Content-Type: multipart/related; boundary="BoundaryStr" type="text/xml"
592 --BoundaryStr
593 Content-Type: text/xml

594 <S11:Envelope
595   xmlns:S11="http://schemas.xmlsoap.org/soap/envelope/"
596   xmlns:wssse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
597   wsswssecurity-secext-1.0.xsd"
598   xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"
599   xmlns:ds="http://www.w3.org/2000/09/xmldsig#"

600   <S11:Header>
601     <wsse:Security>

602       <wsse:BinarySecurityToken wsu:Id="Acert"
603         EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
604   200401-wss-soap-message-security-1.0#Base64Binary"
605         ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-
606   200401-wss-x509-token-profile-1.0#x509v3">
607         ...
608       </wsse:BinarySecurityToken>

609     <xenc:EncryptedKey Id='EK'>
610       <EncryptionMethod
611         Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
612       <ds:KeyInfo Id="keyinfo">
613         <wsse:SecurityTokenReference>
614           <ds:X509Data>
615             <ds:X509IssuerSerial>
616               <ds:X509IssuerName>
617                 DC=ACMECorp, DC=com
618               </ds:X509IssuerName>
619               <ds:X509SerialNumber>12345678</X509SerialNumber>
620             </ds:X509IssuerSerial>
621           </ds:X509Data>
622         </wsse:SecurityTokenReference>
623       </ds:KeyInfo>
624       <CipherData><CipherValue>xyzabc</CipherValue></CipherData>
625       <ReferenceList>
626         <DataReference URI='#EA' />

```

```

627     <DataReference URI='#ED' />
628     </ReferenceList>
629     </EncryptedKey>

630     <xenc:EncryptedData
631       Id='EA'
632       Type="http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-
633 profile-1.0#Attachment-Content-Only"
634       MimeType="image/png">
635       <xenc:EncryptionMethod
636         Algorithm='http://www.w3.org/2001/04/xmlenc#aes128-cbc' />
637       <xenc:CipherData>
638         <xenc:CipherReference URI=cid:bar">
639           <xenc:Transforms>
640             <ds:Transform Algorithm="http://docs.oasis-
641 open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-
642 Only-Transform"/>
643           </xenc:Transforms>
644         </xenc:CipherReference>
645       </xenc:CipherData>
646     </xenc:EncryptedData>

647   </wsse:Security>
648 </S11:Header>
649 <S11:Body>
650   <xenc:EncryptedData Id='ED'
651     <xenc:EncryptionMethod
652       Algorithm='http://www.w3.org/2001/04/xmlenc#aes128-cbc' />
653     <xenc:CipherData>
654       <xenc:CipherValue>DEADBEEF</xenc:CipherValue>
655     </xenc:CipherData>
656   </xenc:EncryptedData>
657 </S11:Body>
658 </S11:Envelope>
659 --BoundaryStr
660 Content-Type: application/octet-stream
661 Content-ID: <bar>
662 Content-Transfer-Encoding: binary

663 BinaryCipherData

```

## 664 4.6 Signing and Encryption

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

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

---

## 5 References

678

- 679 **[CHARSETS]** Character sets assigned by IANA. See <ftp://ftp.isi.edu/in->  
680 [notes/iana/assignments/character-sets](ftp://ftp.isi.edu/in-notes/iana/assignments/character-sets).
- 681 **[DecryptT]** M. Hughes et al, "Decryption Transform for XML Signature", W3C Recommendation, 10  
682 December 2002. <http://www.w3.org/TR/xmlenc-decrypt/>.
- 683 **[Excl-Canon]** Exclusive XML Canonicalization, Version 1.0, W3C Recommendation, 18 July 2002.  
684 <http://www.w3.org/TR/xml-exc-c14n/>.
- 685 **[MTOM]** SOAP Message Transmission Optimization Mechanism, W3C Recommendation, 25  
686 January 2005, <http://www.w3.org/TR/soap12-mtom/>.
- 687 **[RFC2045]** Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message  
688 Bodies, IETF RFC 2045, November 1996, <http://www.ietf.org/rfc/rfc2045.txt>.
- 689 **[RFC2046]** Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types, IETF RFC 2046,  
690 November 1996, <http://www.ietf.org/rfc/rfc2046.txt>.
- 691 **[RFC2047]** Multipurpose Internet Mail Extensions (MIME) Part Three: Message Header Extensions  
692 for Non-ASCII Text, IETF RFC 2047, November 1996, <http://www.ietf.org/rfc/rfc2047.txt>.
- 693 **[RFC2048]** Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures,  
694 <http://www.ietf.org/rfc/rfc2048.txt>.
- 695 **[RFC2049]** Multipurpose Internet Mail Extensions(MIME) Part Five: Conformance Criteria and  
696 Examples, <http://www.ietf.org/rfc/rfc2049.txt>.
- 697 **RFC2119]** S. Bradner, Key words for use in RFCs to Indicate Requirement Levels, IETF RFC 2119,  
698 March 1997, <http://www.ietf.org/rfc/rfc2119.txt>.
- 699 **[RFC2392]** E. Levinson, *Content-ID and Message-ID Uniform Resource Locators*, IETF RFC 2392,  
700 <http://www.ietf.org/rfc/rfc2392.txt>
- 701 **[RFC2557]** MIME Encapsulation of Aggregate Documents, such as HTML (MHTML), IETF RFC  
702 2557, March 1999, <http://www.ietf.org/rfc/rfc2557.txt>.
- 703 **[RFC2633]** Ramsdell B., "S/MIME Version 3 Message Specification", Standards Track RFC 2633,  
704 June 1999. <http://www.ietf.org/rfc/rfc2633.txt>
- 705 **[RFC2822]** Internet Message Format, IETF RFC 2822, April 2001, <http://www.ietf.org/rfc/rfc2822.txt>.
- 706 **[SECGLO]** Informational RFC 2828, "[Internet Security Glossary](http://www.ietf.org/rfc/rfc2828.txt)," May 2000.
- 707 **[SOAP11]** W3C Note, "[SOAP: Simple Object Access Protocol 1.1](http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211)," 08 May 2000.
- 708 **[SwA]** W3C Note, "SOAP Messages with Attachments", 11 December 2000,  
709 <http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211> .
- 710 **[URI]** T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic  
711 Syntax," [RFC 2396](http://www.ietf.org/rfc/rfc2396.txt), MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998,  
712 <http://www.ietf.org/rfc/rfc2396.txt>.
- 713 **[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)  
714 [i.org/Profiles/AttachmentsProfile-1.0.html](http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html)
- 715 **[WSS-Sec]** A. Nadalin et al., Web Services Security: SOAP Message Security 1.0 (WS-Security  
716 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)  
717 [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)
- 718 **[XML-Schema]** W3C Recommendation, "XML Schema Part 1: Structures," 2 May 2001,  
719 <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>.  
720 W3C Recommendation, "XML Schema Part 2: Datatypes," 2 May 2001,  
721 <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>.
- 722 **[XML-Sig]** W3C Recommendation, "XML-Signature Syntax and Processing", 12 February 2002,  
723 <http://www.w3.org/TR/xmlsig-core/>.

724 **[XPath]**  
725

W3C Recommendation, "[XML Path Language](http://www.w3.org/TR/xpath)", 16 November 1999,  
<http://www.w3.org/TR/xpath>.

---

## 726 A. Acknowledgments

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

- 729 • Gene Thurston, AmberPoint
- 730 • Frank Siebenlist, Argonne National Laboratory
- 731 • Hal Lockhart, BEA Systems, Inc.
- 732 • Corinna Witt, BEA Systems, Inc.
- 733 • Thomas DeMartini, ContentGuard
- 734 • Guillermo Lao, ContentGuard
- 735 • Merlin Hughes, Cybertrust
- 736 • Sam Wei, Documentum
- 737 • Tim Moses, Entrust
- 738 • Carolina Canales-Valenzuela, Ericsson
- 739 • Dana Kaufman, Forum Systems, Inc.
- 740 • Toshihiro Nishimura, Fujitsu
- 741 • Kefeng Chen, GeoTrust
- 742 • Irving Reid, Hewlett-Packard
- 743 • Kojiro Nakayama, Hitachi
- 744 • Paula Austel, IBM
- 745 • Derek Fu, IBM
- 746 • Maryann Hondo, IBM
- 747 • Kelvin Lawrence, IBM
- 748 • Michael McIntosh, IBM
- 749 • Anthony Nadalin, IBM
- 750 • Nataraj Nagaratnam, IBM
- 751 • Ron Williams, IBM
- 752 • Ramanathan Krishnamurthy, IONA
- 753 • Kate Cherry, Lockheed Martin
- 754 • Paul Cotton, Microsoft Corporation
- 755 • Vijay Gajjala, Microsoft Corporation
- 756 • Martin Gudgin, Microsoft Corporation
- 757 • Chris Kaler, Microsoft Corporation
- 758 • Rich Levinson, Netegrity, Inc.
- 759 • Jeff Hodges, Neustar, Inc.

- 760 • Frederick Hirsch, Nokia
- 761 • Abbie Barbir, Nortel Networks
- 762 • Lloyd Burch, Novell
- 763 • Steve Anderson, OpenNetwork
- 764 • Vamsi Motukuru, Oracle
- 765 • Ramana Turlapati, Oracle
- 766 • Chong-Jen Hsu, PeopleSoft
- 767 • Prateek Mishra, Principal Identity
- 768 • Ben Hammond, RSA Security
- 769 • Rob Philpott, RSA Security
- 770 • Martijn de Boer, SAP
- 771 • Blake Dournaee, Sarvega
- 772 • Coumara Radja, Sarvega
- 773 • Pete Wenzel, SeeBeyond Technology Corporation
- 774 • Ronald Monzillo, Sun Microsystems
- 775 • Jan Alexander, Systinet
- 776 • Symon Chang, Tibco
- 777 • John Weiland, US Dept of the Navy
- 778 • Phillip Hallam-Baker, Verisign
- 779 • 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. ( <a href="http://www.oasis-open.org/archives/wss/200408/msg00046.html">http://www.oasis-open.org/archives/wss/200408/msg00046.html</a> ) 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 ( <a href="http://www.oasis-open.org/archives/wss/200408/msg00048.html">http://www.oasis-open.org/archives/wss/200408/msg00048.html</a> )
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"> <li>1. Only allow CID references for WS-Security references, for simplicity and interoperability.</li> <li>2. Constrain statement on RFC2047 encoding in section 4.4.1, #6.</li> <li>3. Clarify use of &lt;xenc:EncryptedData&gt; MimeType attribute in 4.5.2, #4. (Issue 345, #1)</li> <li>4. Add statement from interop document regarding MIME boundary for primary SOAP envelope</li> <li>5. Editorial changes to make MAY/MUST/SHOULDs capitalized where possible, other editorial fixes.</li> </ol>

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 &lt;&gt; 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
16	03/07/05	Frederick Hirsch	<p>(Line numbers for diff version)</p> <p>Add Exclusive Canonicalization (691-2) and XML Signature references.(731-2)</p> <p>Issue 349 resolution: Revised language to SHOULD NOT for Reference List lines 506, 568-9. Typo resolution line 199, 303.</p> <p>Issue 356 resolution: typos 199, 540, Change MTOM reference to W3C Recommendation. (693-5)</p> <p>Address public review comments in message <a href="http://www.oasis-open.org/apps/org/workgroup/wss/email/archives/200502/msg00054.html">http://www.oasis-open.org/apps/org/workgroup/wss/email/archives/200502/msg00054.html</a> )</p> <p>1 add goals 84-103</p> <p>2 109-114</p> <p>3 section 2, 157-191</p> <p>4 sec 4.3, 270</p> <p>5 sec 4.4.4, 415-6</p> <p>6 sec 1 90-91, sec 3 198-212, 4.4.2 - 378-386, 4.4.4 405, 4.4.5, 429-430</p> <p>7 sec 1, 109-120</p> <p>9 out of scope sec 1, 98-99</p> <p>10 out of scope, sec 1 100-103</p> <p>11 sec 4.5.2, 540-543</p>
17	03/16/05	Frederick Hirsch	Do not require exclusive canonicalization of attachments, back to what cd-01 said, with minor editorial changes. This means there are no substantive changes since completion of public review of cd-01.
18	04/21/05	Frederick Hirsch	Added text to 4.3.1 and 4.3.2 to resolve issue 376 – defining input and output octet streams of Reference Transforms.

<b>Rev</b>	<b>Date</b>	<b>By Whom</b>	<b>What</b>
19	04/16/05	Frederick Hirsch	Formatting update to changes in draft 18 (editorial). Changes to address issue 377 (use of ReferenceList) – last paragraph in 4.5 (before 4.5.1) and #8 in 4.5.2.

## C. Notices

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

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

793 **Copyright © OASIS Open 2005. All Rights Reserved.**

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

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

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

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