Web Services Security
SOAP with Attachments (SwA) Profile
1.0

OASIS Draft 34, 2812 May-Jun 2004

Document identifier:
ws-swa-profile-1.0-draft-034

Location:

Editors:
Frederick Hirsch, Nokia
TBD

Contributors:
Thomas DeMartini, ContentGuard
Dale Moberg, Cyclone Commerce
Frederick Hirsch, Nokia
Michael McIntosh, IBM
Jerry Schwarz, Oracle

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

Status:
This is a Draft proposal and has no standing.
Committee members should submit comments and potential errata to the ws@lists.oasis-open.org list. Others should submit them to the 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 with "subscribe" in the body) or use other OASIS-supported means of submitting comments. The committee will publish vetted errata on the WSS TC web page (http://www.oasis-open.org/committees/wss/).

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).
# Table of Contents

1 Introduction..................................................................................................................................................3

   1.1 Notations and Terminology..................................................................................................................3

      1.1.1 Notational Conventions..................................................................................................................3

      1.1.2 Namespaces.....................................................................................................................................3

      1.1.3 Acronyms and Abbreviations........................................................................................................4

2 Securing SOAP With Attachments...............................................................................................................5

   2.1 Signatures..............................................................................................................................................5

      2.1.1 MIME Part Signature Transforms....................................................................................................6

         2.1.1.1 Attachment-Complete...................................................................................................................6

         2.1.1.2 Attachment-Content-Only.............................................................................................................6

         2.1.1.3 All-Attachments-Complete.........................................................................................................6

   2.2 Encryption............................................................................................................................................7

      2.2.1 Encryption Processing Rules...........................................................................................................7

      2.2.2 Decryption Processing Rules.............................................................................................................7

      2.2.3 Example...........................................................................................................................................8

   2.3 Signing and Encryption..........................................................................................................................8

3 References..................................................................................................................................................11
1 Introduction

This document describes how to use the Web Services Security (WSS): SOAP Message Security (WSS-Sec) standard with SOAP 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 section is non-normative.

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 WSS: 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/>).

Commonly used security terms are defined in the Internet Security Glossary [SECGL]. Readers are presumed to be familiar with the terms in this glossary as well as the definition in the Web Services Security specification.

1.1.2 Namespaces

Namespace URIs (of the general form "some-URI") represents some application-dependent or context-
dependent URI as defined in RFC 2396 [URI]. This specification is designed to work with the general SOAP [SOAP11, SOAP12] message structure and message processing model, and should be applicable to any version of SOAP. The current SOAP 1.1 namespace URI is used herein to provide detailed examples, but there is no intention to limit the applicability of this specification to a single version of SOAP.

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>S11</td>
<td><a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a></td>
</tr>
<tr>
<td>S12</td>
<td><a href="http://www.w3.org/2003/05/soap-envelope">http://www.w3.org/2003/05/soap-envelope</a></td>
</tr>
<tr>
<td>wsse</td>
<td><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></td>
</tr>
<tr>
<td>wsu</td>
<td><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></td>
</tr>
<tr>
<td>wsswa</td>
<td><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></td>
</tr>
<tr>
<td>mdcrypt</td>
<td><a href="http://www.w3.org/2002/07/decrypt#MIME">http://www.w3.org/2002/07/decrypt#MIME</a></td>
</tr>
</tbody>
</table>

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

1.1.3 Acronyms and Abbreviations

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

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CID</td>
<td>Content ID scheme for URLs. Refers to Multipart MIME body part, that includes both MIME headers and content for that part. [RFC2392]</td>
</tr>
<tr>
<td>SwA</td>
<td>SOAP with Attachments</td>
</tr>
</tbody>
</table>
2 Securing SOAP With Attachments

Attachments may be associated with SOAP messages, as outlined in SOAP With Attachments (SwA). This profile defines how such attachments may be secured for integrity and confidentiality using the OASIS [WSS: Web Services Security: SOAP Message Security] standard. This does not preclude using other techniques such as MTOM as appropriate. The requirements in this section only apply when securing SwA attachments explicitly.

Attachments may be referenced using a CID scheme URL to refer to the attachment that has a Content-ID MIME header value that corresponds to the URL scheme, as defined in [RFC 2392]. This profile is only applicable to SOAP attachments that may be referenced using a CID scheme URL.

Editors Note: Allow URL references corresponding to Content-Location headers, relative and absolute paths? (see http://www.oasis-open.org/archives/wss/200406/msg00012.html)

2.1 Signatures

An attachment may be signed for integrity protection, protecting either the entire MIME part including MIME headers, or only the MIME part content.

As outlined in SOAP Message Security, the <ds:Signature> element is conveyed in the <wsse:Security> SOAP header block. The <ds:Signature> may protect the integrity of an attachment and provide origin authentication by including a <ds:Reference> element that refers to that attachment, as outlined in this profile.

An attachment may only be protected if it includes a Content-ID MIME header. The attachment MUST be referenced using a CID scheme URI as the <ds:Reference> URI attribute value. This URL value MUST correspond to the Content-ID MIME header value.

The <ds:Reference> indicates whether the entire MIME part including MIME headers is to be included in the hash calculation, or only the content of the MIME part. This is done by specifying a "MIME Part Signature Transform". The definition of this transform may also define additional processing rules necessary to prepare the MIME part for the hash calculation.

The "MIME Signature Transform" MUST be specified using a URI as the Algorithm attribute value for a <ds:Transform> element conveyed as the immediate child of the <ds:Transforms> element. The <ds:Transforms> element is the immediate child of the <ds:Reference> element.

The content of a MIME part is signed or verified after the content is decoded, according to the mechanism specified by the Content-Transfer-Encoding MIME header, if present. Thus a change in Content-Transfer-Encoding should not impact signature verification if the recipient supports all the standard encodings defined in RFC 2045.

Example:

```xml
Content-Type: multipart/related; boundary="arggh" type=text/xml
--arggh
Content-Type: text/xml
<S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..." xmlns:ds="..." xmlns:xenc="...">
  <S11:Header>
    <wsse:Security>
      <ds:Signature>
        <ds:Reference URI="cid:bar">
          <ds:Transforms>
            <ds:Transform Algorithm="mime-content-only-URI"/>
          </ds:Transforms>
        </ds:Reference>
      </ds:Signature>
    </wsse:Security>
  </S11:Header>
</S11:Envelope>
```
2.1.1 MIME Part Signature Transforms

Two MIME part signature transforms are defined in this profile, others may also be defined.

Every "MIME Part Signature Transform" MUST canonicalize the MIME part before creating a hash of the portion of the MIME part to be signed, depending on the MIME Type. For example, for MIME parts of type text, the line endings must be canonicalized to <CR><LF> and the charset must be a registered charset (see RFC 2311 section "Canonicalization"). [RFC2311, CHARSETS, RFC2045].

2.1.1.1 Attachment-Complete

This transform should be used when the complete MIME part, including the MIME headers and content, is to be signed, to signal what is signed.

The processor must canonicalize the MIME part before signing and verification as necessary and noted above.

This transform MUST be identified using the URI value: TBD (http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Complete)

2.1.1.2 Attachment-Content-Only

This transform should be used when only the MIME part content is to be signed, to signal what is signed, and must canonicalize the MIME part before signing and verification as necessary and noted above.

The processor must canonicalize the MIME part before signing and verification as necessary and noted above.

This transform MUST be identified using the URI value: TBD (http://docs.oasis-open.org/wss/2004/XX/oasis-2004XX-wss-swa-profile-1.0#Attachment-Content-Only)

2.1.1.3 All-Attachments-Complete

This transform should be used when all the MIME parts including all MIME part content for each is to be signed, to signal what is signed, and must canonicalize each MIME part before signing and verification as necessary and noted above. This is useful for use cases where addition and removal of MIME parts must be prevented, while allowing the primary SOAP part to be processed as expected, regardless of attachments.

Editors Note: (see http://www.oasis-open.org/archives/wss/200406/msg00012.html)
2.2 Encryption

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

This may be done using XML Encryption to encrypt the attachment (either content or entire part including MIME headers), placing the resulting cipher text in the updated attachment body and the <xenc:EncryptedData> element in the <wsse:Security> header. An <xenc:CipherReference> is used to link the cipher data to the <xenc:EncryptedData> element.

No <xenc:ReferenceList> element is placed in the <wsse:Security> header, since the <xenc:EncryptedData> element is present in the header, eliminating the need for a reference. The SOAP Message Security standard recommends the use of <xenc:ReferenceList>, but this is only necessary when the <xenc:EncryptedData> element is not present in the <wsse:Security> header.

2.2.1 Encryption Processing Rules

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

1. Encrypt the attachment part using XML Encryption, according to the rules of XML Encryption. Encrypt either the entire attachment including MIME headers or only the attachment content.

2. Set the <xenc:EncryptedData> Type attribute value to a URI that specifies adherence to his profile and that specifies what was encrypted (MIME content or entire MIME part including headers).

3. Set the <xenc:EncryptedData> MimeType attribute to match the attachment MIME part Content-Type before encryption.

4. Set the <xenc:EncryptedData> Encoding attribute to match the attachment MIME part Content-Transfer-Encoding MIME header before encryption.

5. Set the <xenc:EncryptedData> <xenc:CipherReference> to the CID scheme URL referring to the attachment part Content-ID before encryption. Ensure this MIME header is in the part conveying the cipher data after encryption.

6. Prepend the <xenc:EncryptedData> element to the <wsse:Security> SOAP header block. Do NOT add a <xenc:ReferenceList> element to the SOAP header block, even though recommended by SOAP Message Security.

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

8. Update the attachment MIME part headers, with a MIME Content-Type and Content-Transfer-Encoding appropriate to the format of the cipher data.

2.2.2 Decryption Processing Rules:

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

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

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

1. Extract the cipher text from the attachment referenced by the CID scheme URL specified in the <xenc:CipherReference> URL attribute.

Determine how to do this based on a URI specified as the Algorithm URI attribute value of the...
<ds:Transform> element within the <ds:Transforms> element within the <xenc:CipherReference> element.

This URI is named the “MIME Part Dereference Transform” in this specification and also specifies how the cipher text may need to be transformed according to the MIME part Content-Transfer-Encoding and other factors (eg. First remove all line endings, then decode etc).

2. Decrypt the cipher text using the information present in the appropriate <xenc:EncryptedData> element and possibly other out of band information, according to the XML Encryption Standard.

3. If the <xenc:EncryptedData>Type attribute indicates that the MIME headers were encrypted, then the MIME headers and cipher text content of the attachment part referenced by the CID scheme URL must be replaced by the result of decryption.

4. If the <xenc:EncryptedData>Type attribute indicates that only the content of the MIME part was encrypted, then the cipher text content of the attachment part referenced by the CID scheme URL must be replaced by the result of decryption. In this case the MIME part Content-Type header value MUST be replaced by the <xenc:EncryptedData> MimeType attribute value, and the MIME part Content-Transfer-Encoding header value MUST be replaced by the <xenc:EncryptedData> Encoding attribute value.

2.2.3 Example

Content-Type: multipart/related; boundary="arggh" type=text/xml
--arggh
Content-Type: text/xml
<S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..." xmlns:ds="..." xmlns:xenc="...">
  <S11:Header>
    <wsse:Security>
      <xenc:EncryptedData Id="foo_Part" Type="url-attachment-with-mime-headers" MimeType="image/jpeg" Encoding="base64">
        <ds:KeyInfo>
          <ds:KeyName>someName</ds:KeyName>
        </ds:KeyInfo>
        <xenc:CipherData>
          <xenc:CipherReference URI="cid:foo">
            <Transforms>
              <ds:Transform Algorithm="wsse:CidCipherTextToMimePartWithOutMimeHeaders"/>
            </Transforms>
          </xenc:CipherReference>
        </xenc:CipherData>
      </xenc:EncryptedData>
    </wsse:Security>
  </S11:Header>
  <S11:Body>
    some information
  </S11:Body>
</S11:Envelope>
--arggh
Content-Type: something
Content-Id: <foo>
Content-Transfer-Encoding: base64
DEADBEEF

2.3 Signing and Encryption

The Decryption Transform for XML Signature was created so that when XML Signature and XML Encryption are combined in one document, a receiver can explicitly determine which decryptions are necessary before signature verification [DecryptT]. A similar problem can occur with attachments when some attachments are signed and some encrypted – a receiver needs to know which attachments to decrypt before verifying signatures.
Thus a new Decryption Transform mode is required for MIME attachments, allowing CID scheme references for attachments containing cipher data.

Consider the following SOAP message with a single attachment, that was both signed and encrypted. How is the receiver to know whether to decrypt before verifying the signature? (Note that this only becomes more complicated with multiple attachments, signatures and encryptions).

```
---arggh
Content-Type: multipart/related; boundary="arggh" type=text/xml
--arggh
Content-Type: text/xml
<SOAP-Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..."
xmlns:ds="..." xmlns:xenc="...">
  <!-- S11:Header -->
  <xenc:EncryptedData Id="foo_Part" Type="url-attachment-with-mimeheaders" MimeType="image/jpeg" Encoding="base64">
    <ds:KeyInfo>
      <ds:KeyName>someName</ds:KeyName>
    </ds:KeyInfo>
    <xenc:CipherData>
      <xenc:CipherReference URI="cid:foo">
        <Transforms>
          <ds:Transform Algorithm="wsse:CidCipherTextToMimePartWithOutMimeHeaders"/>
        </Transforms>
        <ds:Transform Algorithm="mime-content-only-URI"/>
        <ds:Transform Algorithm="attachment-decrypt-URI#Binary"/>
      </Transforms>
      <ds:Signature>
        <ds:Reference URI="cid:foo">
          <ds:Transforms>
            <ds:Transform Algorithm="mime-content-only-URI"/>
            <ds:Transform Algorithm="attachment-decrypt-URI#Binary"/>
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
          <ds:DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk="</ds:DigestValue>
        </ds:Reference>
      </ds:Signature>
      <S11:Body>
        some items
      </S11:Body>
    </S11:EncryptedData>
  </S11:Header>
  <S11:Header>
    <wsse:Security>
      <ds:Signature>
        <ds:Reference URI="cid:foo">
          <ds:Transforms>
            <ds:Transform Algorithm="mime-content-only-URI"/>
            <ds:Transform Algorithm="attachment-decrypt-URI#Binary"/>
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
          <ds:DigestValue>j6lwx3rvEPO0vKtMup4NbeVu8nk="</ds:DigestValue>
        </ds:Reference>
      </ds:Signature>
      <S11:Body>
        some items
      </S11:Body>
    </S11:EncryptedData>
  </S11:Header>
</SOAP-Envelope>
```

Example

```
---arggh
Content-Type: multipart/related; boundary="arggh" type=text/xml
--arggh
Content-Type: text/xml
```

Note: Full example to be updated once processing rules are agreed.
One solution might be to change Content-ID values upon encryption, but this might introduce other problems, like CID scheme references within a message with no apparent corresponding attachment, which might be considered an error early in the receiver processing.

Another solution might be to require an `<xenc:EncryptedData>` element to be in the encrypted attachment itself, clearly signalling the use of encryption. This, however, would require XML processing of both the primary SOAP part as well as the attachment, while placing only cipherdata in the attachment limits this processing to the header.

Thus, a new mode of the Decryption Transform for XML Signature is proposed, for MIME attachments, as shown in this example to indicate that what is signed is indeed the cipherdata. If the mdcrt:Except were not present, then the attachment should be decrypted first.
3 References

The following are normative references:


A. Acknowledgments

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

- TBD
## B. Revision History

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>By Whom</th>
<th>What</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>05/25/04</td>
<td>Frederick Hirsch</td>
<td>Initial version, put draft proposal into profile format.</td>
</tr>
<tr>
<td>2</td>
<td>05/26/04</td>
<td>Frederick Hirsch</td>
<td>Editorial and namespace suggestions from Michael McIntosh. Added rationale for SwA support to introduction. Completely rewrote processing rules for encryption and decryption.</td>
</tr>
<tr>
<td>3</td>
<td>05/28/04</td>
<td>Frederick Hirsch</td>
<td>Rewrote signature section, fixed cid references and Content-Ids, added examples.</td>
</tr>
<tr>
<td>4</td>
<td>06/12/04</td>
<td>Frederick Hirsch</td>
<td>Added Decrypt Transform section, added All-Attachments-Complete transform, changed MIME reference to v3, minor editorial changes.</td>
</tr>
</tbody>
</table>
C. Notices

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

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

Copyright © OASIS Open 2004. All Rights Reserved.

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

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

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

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